

ارائه یک راه کار مبتنی بر یادگیری عمیق برای

تشخیص بدافزارهای اندرویدی

علی علیائی طرهبه^۱، عباس رسولزادگان^{۲*}

دانشجوی دکترای مهندسی کامپیوتر - نرم افزار، گروه مهندسی کامپیوتر، دانشکده مهندسی، دانشگاه فردوسی مشهد، مشهد، ایران^۱

دانشیار گروه مهندسی کامپیوتر، دانشکده مهندسی، دانشگاه فردوسی مشهد، مشهد، ایران^{۲*}



چکیده

اندروید، سیستم عاملی متن باز با سهم بازار گسترده، یکی از اهداف اصلی توسعه دهندگان بدافزار است. افزایش تنوع دستگاه‌های مبتنی بر اندروید و پیچیدگی روزافزون بدافزارها، شناسایی آن‌ها را به چالشی جدی تبدیل کرده است. در این پژوهش، روشی مبتنی بر یادگیری عمیق برای تشخیص بدافزارها ارائه شده است که از تحلیل ایستا بهره می‌برد؛ این روش با تبدیل بایت‌کدهای خام برنامه‌ها به سیگنال‌های صوتی، استخراج اطلاعات مهم از این سیگنال‌ها و آموزش یک مدل با کمک یادگیری انتقالی، به صحت ۹۹.۳ درصد و دقت ۹۹.۸ درصد رسید. با تکیه بر نگاشت بایت‌کدهای خام به حوزه صوت، علاوه بر ایجاد دیدگاه جعبه سیاه گونه، پیچیدگی محاسباتی کاهش و دقت شناسایی افزایش پیدا کرد. یکی از نوآوری‌های این روش که آن را برای استفاده در کاربردهای صنعتی و بازارهای اندرویدی مناسب می‌سازد، قابلیت آن در تشخیص بدافزارهای مبهم شده و عملکرد موفق آن در این زمینه است؛ موضوعی که یکی از چالش‌های اساسی در حوزه تشخیص بدافزارهای اندرویدی به شمار می‌رود.

واژگان کلیدی: بدافزارهای اندروید، یادگیری عمیق، تحلیل ایستا، پردازش صوت، پردازش تصویر.

A Deep Learning Based Method for Android Malware Detection

Ali Olyaei Torqabeh¹, Abbas Rasoolzadegan^{2*}

PhD Student of Software Engineering, Department of Computer Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran¹

Associate Professor of Software Engineering, Department of Computer Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran^{2*}

Abstract

Android malware remains a significant cybersecurity concern, with McAfee's 2022 report calling it "The Year of Stealthy Mobile Attacks." The open-source nature of Android and the widespread use of third-party app sources make it a prime target for cybercriminals. Malware variants such as Trojans, spyware, and worms often disguise themselves as legitimate applications while secretly executing malicious activities. Attackers employ techniques like repackaging apps, injecting harmful updates, and distributing deceptive downloads to infiltrate devices. In 2023 alone, over 600 million malware-infected applications were downloaded from Google Play, underscoring the urgent need for more sophisticated detection strategies. Traditional signature-based methods struggle against rapidly evolving threats, necessitating the adoption of more adaptive and intelligent approaches such as deep learning.

This research presents an innovative Android malware detection framework utilizing static analysis. Instead of relying on conventional feature extraction techniques, this approach transforms raw APK byte sequences into audio signals, allowing the application of spectral analysis methods. The conversion process begins by interpreting the raw byte sequence of an APK file as a one-dimensional numerical signal, where each byte value (0-255) is mapped to an amplitude level. To avoid loss of structural information, the sequence is reshaped and passed through a waveform generation process, effectively converting the byte stream into an audio signal. This transformation preserves the sequential dependencies inherent in the bytecode, allowing for effective feature extraction in the frequency domain. Once the APK file is represented as an audio waveform, cepstral analysis techniques such as Mel-Frequency Cepstral Coefficients (MFCCs), Bark-Frequency Cepstral Coefficients (BFCCs), and Gammatone-Frequency Cepstral Coefficients (GFCCs) are applied. These spectral features capture

* Corresponding author

* نویسنده عهده دار مکاتبات



unique patterns embedded in the bytecode, revealing underlying structures that distinguish malicious applications from benign ones. The extracted features are then formatted as three-channel spectrogram-like images, forming the input to a deep learning model. By leveraging transfer learning on these spectral representations, the model can efficiently classify malware while remaining robust against code obfuscation and adversarial attacks.

To optimize malware classification, this research fine-tunes EfficientNetV2B0, a state-of-the-art convolutional neural network, using transfer learning. The pre-trained model, originally designed for large-scale image classification, is adapted to process spectral representations of APK files. The final layers are retrained on the malware dataset while earlier layers retain their learned feature extraction capabilities. This approach enables efficient learning with limited data while improving generalization, ensuring high detection accuracy even on previously unseen malware samples.

The proposed model achieves an impressive 99.3% accuracy and 99.1% recall, demonstrating its effectiveness in detecting malware, including obfuscated and evasive variants. By preserving the sequential structure of APK bytecode in an audio format, this method enhances detection robustness without requiring complex reverse engineering techniques. The model's performance was evaluated using a diverse dataset of over 11,000 applications, including obfuscated malware samples designed to bypass traditional security measures. The results highlight the effectiveness of the audio-based approach in distinguishing malicious patterns, achieving a detection rate significantly higher than conventional static analysis techniques. Unlike traditional machine learning models that require extensive feature engineering, this black-box method leverages deep learning to automatically extract relevant patterns from spectral features, improving both efficiency and scalability. The high accuracy and recall scores achieved in this study mark a significant advancement in Android malware detection, proving that audio-based representations can effectively capture the structural and behavioral properties of malicious software. This research not only provides a novel direction for malware classification but also introduces a scalable and adaptable framework capable of addressing modern threats with unprecedented accuracy and resilience.

Keywords: Android Malware Detection, Deep Learning, Static Analysis, Image Processing, Audio Signal Processing.

همچنین، در سال ۲۰۲۳ بیش از ششصد میلیون بار بدافزارها از طریق گوگل پلی بارگیری شده‌اند.^۲ این آمارها نشان‌دهنده نیاز فوری به روش‌های کارآمد برای شناسایی و مقابله با بدافزارها است.

در سال‌های اخیر، روش‌های تشخیص مبتنی بر امضا به دلیل ناتوانی در شناسایی بدافزارهای جدید یا مبهم‌سازی‌شده، ناکارآمد شده‌اند [۶]. در پاسخ به این چالش، رویکردهای نوین مبتنی بر یادگیری عمیق و تحلیل‌های غیرسنتی مانند تبدیل کدهای اجرایی به سیگنال‌های صوتی یا تصاویر، محبوبیت یافته‌اند. این روش‌ها بدون نیاز به مهندسی معکوس و تجزیه کد قادرند الگوهای پنهان را از طریق ویژگی‌های استخراج‌شده شناسایی کنند.

انگیزه اصلی این پژوهش، ارائه راه‌کاری نوآورانه برای تشخیص بدافزارهای اندرویدی با استفاده از یادگیری عمیق و بهره‌گیری از اطلاعات ساختاری بایت‌کدها بدون نیاز به تجزیه برنامه‌ها است؛ بدین منظور، بایت‌کد خام برنامه‌ها به سیگنال‌های صوتی تبدیل و سه نوع طیف MFCC، BFCC و GFCC از آن استخراج شده‌اند. این طیف‌ها به صورت تصویر رنگی سه‌کاناله تبدیل شده‌اند تا با کمک یادگیری انتقالی، مدل از پیش آموزش‌دیده EfficientNetV2B0 آموزش ببیند تا بتواند طبقه‌بندی نهایی را انجام دهد. از مزایای مهم این روش می‌توان به مقاومت بالا در برابر مبهم‌سازی کد، کاهش نیاز به مهندسی معکوس، کارایی بالا در محیط‌های واقعی و صنعتی و قابلیت اجرا

۱- مقدمه

با افزایش تولید و انتشار بدافزارهای اندرویدی، نگرانی‌ها درباره امنیت کاربران این اکوسیستم به شکل چشم‌گیری افزایش یافته است. گزارش اخیر McAfee سال ۲۰۲۲ را «سال حملات مخفیانه تلفن همراه» معرفی کرده است؛ ماهیت متن‌باز سیستم عامل اندروید، نصب آسان از منابع ناشناس و نبود نظارت کامل بر فروشگاه‌های برنامه، این سکو را به هدفی ایدئال برای مهاجمان سایبری تبدیل کرده است [۱].

بدافزارهای اندرویدی با استفاده از روش‌هایی مانند بسته‌بندی مجدد، ارتقای حمله و تغییر ظاهر برنامه، سعی در دورزدن سامانه‌های امنیتی دارند. این بدافزارها با اهدافی نظیر سرقت اطلاعات، شنود، اجرای مخفی فرامین و ایجاد اختلال در عملکرد سامانه طراحی می‌شوند و در قالب انواعی مانند ویروس‌ها، تروجان‌ها، نرم‌افزارهای جاسوسی و کرم‌ها ظاهر می‌شوند [۲، ۵].

آمارها^۱ نشان می‌دهند که حدود هر هفت‌ثانیه یک بدافزار جدید منتشر می‌شود. طبق گزارش‌های McAfee و Kaspersky، کشورهایمانند ایران، الجزایر و نیجریه جزو آلوده‌ترین کشورها هستند؛ برای مثال، ۳۵ درصد از تلفن‌های همراه ایرانی، بیشترین میزان آلودگی به بدافزار را در بین تلفن‌های همراه کاربران دیگر کشورها دارند.^۲

¹ <https://www.gdatasoftware.com/news/g-data-mobile-malware-report-2019-new-high-for-malicious-android-apps>

² <https://financialtribune.com/articles/sci-tech/114469/phone-malware-widespread-in-iran>

³ <https://www.kaspersky.com/blog/malware-in-google-play-2023/49579>

در بررسی و تحلیل برنامه‌های اندرویدی، سه رویکرد اصلی وجود دارد: تحلیل ایستا، تحلیل پویا و تحلیل ترکیبی. در تحلیل ایستا، کد برنامه‌ها بدون اجرای آن‌ها بررسی می‌شود. این روش به دلیل امکان دسترسی مستقیم به کد منبع یا محتویات باینری فایل برنامه‌ها و استخراج ویژگی‌هایی مانند رشته‌ها، دسترسی‌ها و برخی توابع، بسیار سریع است؛ با این حال، تحلیل ایستا در مواجهه با روش‌های مبهم‌سازی^۱، مانند تغییر نام متغیرها، کدگذاری رشته‌ها و درج کدهای بی‌استفاده دچار چالش می‌شود؛ از سوی دیگر، تحلیل پویا با اجرای برنامه در محیط کنترل‌شده، رفتار برنامه را در زمان اجرا بررسی می‌کند. این روش قادر است، فعالیت‌های مخرب پنهان‌شده در زمان اجرا، مانند دسترسی غیرمجاز به داده‌ها یا ارتباطات مشکوک شبکه‌ای را شناسایی کند؛ با این حال، تحلیل پویا زمان‌بر بوده و نیازمند منابع بیشتری است. تحلیل ترکیبی با ترکیب مزایای دو روش پیشین، رویکرد جامعی برای بررسی برنامه‌ها ارائه می‌دهد.

یکی از چالش‌های اساسی در این حوزه، استفاده گسترده از روش‌های مبهم‌سازی توسط طراحان بدافزار است که فرایند تشخیص را پیچیده‌تر می‌کند. این روش‌ها به بدافزارها کمک می‌کنند تا از شناسایی به‌وسیله ابزارهای مبتنی بر امضا، جلوگیری کنند؛ در نتیجه، روش‌های سنتی مبتنی بر امضا که بر شناسایی الگوهای ثابت در کدها متکی هستند، دیگر پاسخ‌گوی پیچیدگی بدافزارهای جدید نیستند. در سال‌های اخیر، روش‌های مبتنی بر یادگیری ماشین و یادگیری عمیق به دلیل توانایی بالای آن‌ها در شناسایی الگوهای پنهان و تحلیل داده‌های پیچیده، جایگزین روش‌های سنتی شده‌اند. این رویکردها با استخراج خودکار ویژگی‌ها و ارائه مدل‌های انعطاف‌پذیرتر، نقش مهمی در بهبود دقت و سرعت تشخیص بدافزارها ایفا می‌کنند.

کانفورا و همکاران در [۷] روشی برای شناسایی بدافزارهای اندرویدی ارائه کردند که مبتنی بر محاسبه بافت‌نگاشت فرکانس دستورهای اجرایی کد^۲ و تحلیل آن‌ها است. در این روش، با بررسی فراوانی دستورهای عملیاتی خاص استخراج‌شده از کدهای Dalvik، تشخیص بدافزار انجام می‌شود. این دستورات بر حسب معمول برای تغییر جریان کنترل برنامه استفاده می‌شوند. روش پیشنهادی روی مجموعه‌داده Drebin و با استفاده از چندین طبقه‌بند مبتنی بر ساختار درخت ارزیابی شده و به دقت ۹۵ درصد دست یافته است.

الزیلابی و همکاران در [۸]، روش DL-Droid را معرفی کردند که یک رویکرد مبتنی بر یادگیری عمیق برای شناسایی بدافزارهای اندرویدی با استفاده از تحلیل پویا و تولید ورودی حالت است. در این پژوهش، بیش از ۳۱۰۰۰ برنامه جمع‌آوری

شد که بیش از یازده‌هزار مورد آن‌ها بدافزار بودند. DL-Droid روی یک سکوی خودکار اجرا می‌شود که قادر به انجام تحلیل ایستا و پویا است. برای ارزیابی، از یک دستگاه اندروید واقعی استفاده شد که زمان تحلیل هر برنامه حدود ۱۹۰ ثانیه است. این روش با استفاده از ویژگی‌های تحلیل پویا به نرخ تشخیص ۹۷.۸ درصد و با افزودن ویژگی‌های تحلیل ایستا به ۹۹.۶ درصد دست یافت.

قاسمی و همکاران در پژوهش [۳۳]، ویژگی‌های برنامه‌های اندروید را هنگام نصب و اجرا از سمت تلفن همراه استخراج و برای پردازش و تحلیل به سرورهای ابری ارسال کردند؛ سپس با کمک الگوریتم‌های پایه یادگیری ماشین، تشخیص بدافزار را انجام می‌دهند. این روش به دلیل استفاده از محاسبات ابری، سرعت بالا، دقت مناسب و مصرف منابع کمتر را در پی دارد که بر روی مجموعه‌داده Drebin، به دقت ۹۶.۴۴ درصد دست یافتند. **دی پیر و همکاران** در پژوهش [۳۴]، روشی برای ارزیابی ریسک امنیتی برنامه‌های موبایلی ارائه کرده‌اند که بر مبنای تحلیل مجوزهای درخواستی به‌وسیله برنامه‌ها عمل می‌کند. آن‌ها با بررسی مجوزهای ده‌ها بدافزار و صدها برنامه قانونی، معیاری جدید برای سنجش ریسک امنیتی طراحی کردند و یک ابزار نرم‌افزاری برای پیاده‌سازی آن توسعه داده‌اند.

پژوهش‌گران در پژوهش [۹] با استخراج گراف فراخوانی‌های API‌های سامانه‌ای و استفاده از معماری مبتنی بر شبکه CNN، تشخیص بدافزار را انجام داده‌اند. دقت تشخیص بدافزار در این پژوهش ۹۹ درصد و برای بدافزارهای جدید ۹۱ درصد گزارش شد.

در پژوهش [۱۰]، شی-سیائو و همکاران با استفاده از توالی فراخوانی‌های سامانه‌ای در حین اجرای برنامه، دو شبکه حافظه کوتاه‌مدت ماندگار^۳ را آموزش دادند؛ یکی با برنامه‌های خوش‌خیم و دیگری با بدافزارها. تشخیص بدافزار با مقایسه امتیازهای این دو شبکه و به‌وسیله یک طبقه‌بند انجام می‌شود. این پژوهش روی مجموعه‌داده Derbin و حدود هفت‌هزار برنامه ارزیابی شد و به دقت ۹۶ درصد دست یافت.

در [۱۱]، روش DeepAMD معرفی شد که ترکیبی از تحلیل ایستا و پویا است. در تحلیل ایستا، دقت تشخیص بدافزار ۹۳ درصد، تشخیص دسته بدافزار ۹۲ درصد و تشخیص خانواده بدافزار نود درصد بود. در تحلیل پویا، دقت تشخیص دسته بدافزار هشتاد درصد و خانواده بدافزار ۵۹ درصد بود. این پژوهش از مجموعه‌داده CICInvesAndMal2019 استفاده کرد و از طبقه‌بندهایی مانند بیز ساده^۴، شبکه عصبی چندلایه^۵ و درخت تصمیم^۶ بهره برد.

^۳ Long Short-Term Memory (LSTM)

^۴ Naive Bayes

^۵ Multi-Layer Perceptron

^۶ Decision Tree

^۱ Obfuscation

^۲ Opcode

به دقت ۹۴ درصد رسید. در پژوهش دیگری کاسولار و همکاران در [۲۳] با تبدیل بایت کدها به فایل های صوتی، بر تشخیص خانواده بدافزار متمرکز شدند. این پژوهش از ویژگی هایی مانند نرخ عبور از صفر^۷ و MFCC استفاده کرد و به دقت ۹۸ درصد دست یافت.

پژوهش گران در [۲۴] با ترکیب پردازش تصویر و صوت، بایت کدهای APK را به فایل های صوتی WAV تبدیل کردند و با استفاده از شبکه های CNN به دقت ۹۶ درصد در تشخیص بدافزار و نود درصد در تشخیص خانواده بدافزار رسیدند. پاول و همکاران در [۲۵] با استخراج بیش از صد ویژگی از تبدیل بایت کدهای APK به WAV، روشی برای تشخیص بدافزار ارائه کردند. این پژوهش روی مجموعه داده CICMalDroid 2020 انجام شد و به دقت ۹۸٫۹ درصد دست یافت.

با بررسی پژوهش های مرتبط مشخص می شود که تاکنون روش های متنوعی برای نگاشت بایت کدهای برنامه های اندرویدی به حوزه های تصویری یا صوتی پیشنهاد شده اند. برخی از این روش ها تنها از یک نوع طیف صوتی مانند MFCC یا ویژگی های تصویری خاکستری استفاده کرده اند که اطلاعات ساختاری محدودی ارائه می دهند؛ همچنین بسیاری از این پژوهش ها نیازمند طراحی و آموزش مدل های اختصاصی از ابتدا بوده اند که فرآیند یادگیری را زمان بر و وابسته به منابع محاسباتی بالا کرده است.

در این میان، شکاف مهمی در استفاده ترکیبی از چندین طیف از داده های صوتی و استفاده از آن ها به صورت تصویر رنگی سه کاناله وجود دارد؛ همچنین، استفاده از مدل های پیش آموزش دیده بهینه مانند EfficientNetV2B0، که علاوه بر سبک بودن، دقت بالایی نیز دارند و روش هایی مانند یادگیری انتقالی برای آموزش مدل ها، کمتر مورد توجه قرار گرفته است؛ بنابراین نوآوری اصلی پژوهش حاضر را می توان در سه بُعد تشریح کرد:

۱. ترکیب سه طیف صوتی MFCC، BFCC و GFCC برای ایجاد تصویری غنی تر و سه کاناله از برنامه ها؛
۲. استفاده از این تصویر رنگی به عنوان ورودی مدل از پیش آموزش دیده EfficientNetV2B0 در قالب یادگیری انتقالی؛
۳. ارزیابی دقیق مدل روی داده های واقعی شامل نمونه های مبهم سازی شده، بدون نیاز به استخراج دستی ویژگی یا مهندسی معکوس.

این ترکیب از نگاشت بایت کد به حوزه صوت، ایجاد تصویر رنگی و استفاده از یادگیری انتقالی روی مدلی سبک و بهینه، تاکنون در پژوهش های موجود به صورت یک پارچه انجام نشده و پژوهش حاضر با هدف پرکردن این شکاف و ارائه روشی دقیق، سریع و مقاوم در برابر مبهم سازی طراحی شده است.

⁷ Zero-Crossing Rate (ZCR)

طهرانی و همکاران در [۱۲] با استفاده از الگوریتم های بردار پشتیبان ماشین^۱ و نزدیک ترین همسایه^۲، روشی برای تشخیص بدافزار ارائه کردند که بر بهبود دقت و کاهش نرخ خطا متمرکز است؛ این پژوهش همچنین به جمع آوری و متعادل سازی مجموعه داده پرداخت. با استفاده از ویژگی هایی مانند فهرست دسترسی ها و سرویس ها، دقت نهایی ۹۹ درصد گزارش شد.

لیمین شن در [۱۳] از شبکه حافظه کوتاه مدت ماندگار پیچشی مبتنی بر ویژگی خود آگاهی^۳ برای تشخیص بدافزارهای اندرویدی استفاده کرد. در این پژوهش، داده های شبکه ای برنامه ها به تصاویر سطوح خاکستری تبدیل شدند و به عنوان ورودی شبکه عصبی عمیق استفاده شدند. دقت تشخیص بدافزار ۹۵ درصد و تخمین خانواده بدافزار ۷۲ درصد بود.

باکور در [۱۴] روش DeepVisDroid را معرفی کرد که با تبدیل سورس کد نرم افزار کاربردی به تصاویر سطوح خاکستری^۴، چهار مجموعه داده تصویری تولید کرده و با کمک الگوریتم های SIFT، SURF، ORB و KAZE استخراج ویژگی ها انجام شده است. این روش روی مجموعه داده های Derbin و MalGenome ارزیابی شد و به دقت ۹۸ درصد دست یافت.

یوژین دینگ و همکاران در [۱۵] با تبدیل بایت کدهای فایل APK به تصاویر سطوح خاکستری، تشخیص بدافزار را انجام دادند. این پژوهش از مجموعه داده Derbin استفاده کرد و با معماری CNN به دقت ۹۴ درصد رسید؛ همچنین دنیش وسن و همکاران در [۱۷] از تصاویر رنگی RGB برای تشخیص بدافزار استفاده کردند. در روش IMCFN، از تنظیم مجدد^۵ شبکه های معروفی مانند InceptionNet استفاده شد. این پژوهش روی مجموعه داده های IoT-android mobile و Malimg malware ارزیابی شد. در [۱۸] نیز با تبدیل بایت کدها به تصاویر خاکستری و ارزیابی پنجاه هزار برنامه، دقت ۹۴ درصد به دست آمد. در پژوهش دیگری پوجا یاداو و همکاران در [۱۹] با تبدیل فایل های دودویی به تصاویر رنگی، روشی برای تشخیص بدافزار ارائه کردند. این پژوهش از معماری EfficientNet استفاده کرد و به دقت ۹۵ درصد دست یافت.

فرخ منش و همکاران در [۲۱] با استفاده از روش های استخراج اطلاعات از موسیقی^۶، فایل های APK را به فایل های MIDI و سپس WAV تبدیل کردند و ویژگی های متنوعی را که از یک فایل صوتی قابل دریافت است، استخراج کردند؛ همچنین فرانچسکو مرکالدو و همکاران در [۲۲] با تبدیل بایت کدهای فایل Classes.dex به فایل های صوتی WAV، بررسی کردند که آیا بدافزارهای مشابه نمونه های صوتی مشابهی دارند یا خیر. این پژوهش از ویژگی هایی مانند کروماگرام و MFCC استفاده کرد و

¹ Support Vector Machine (SVM)

² K-Nearest Neighbors (KNN)

³ Self-Attention

⁴ Grayscale

⁵ Fine-tune

⁶ Music Information Retrieval (MIR)



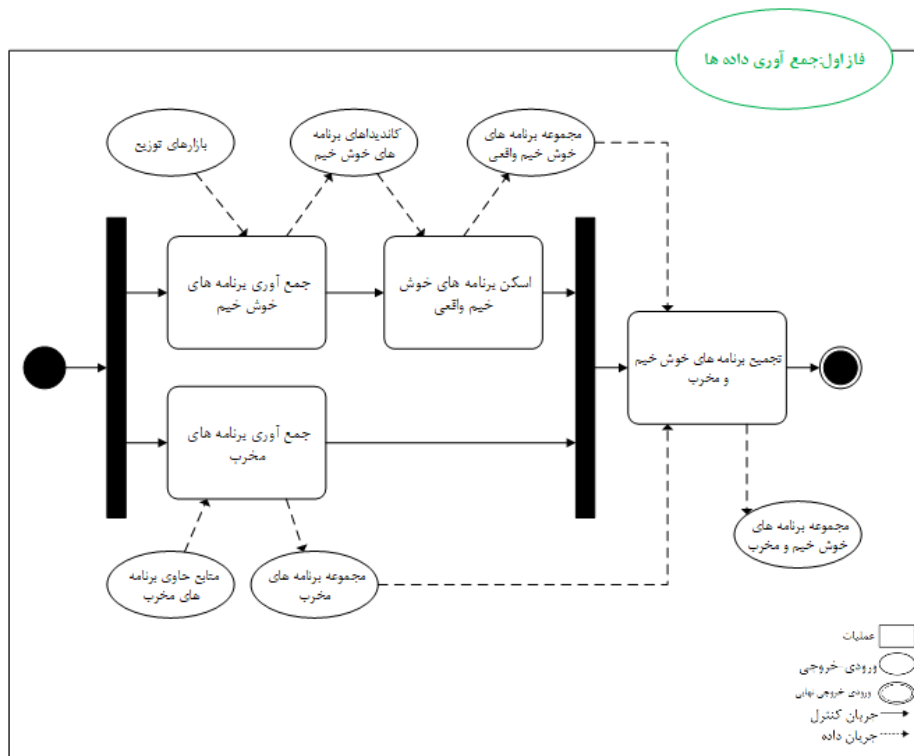
۳- روش پیشنهادی

پژوهش حاضر با استفاده از رویکرد تحلیل ایستا برای تشخیص بدافزارها انجام شده است. در تحلیل ایستا، داده‌های خام بایت‌کدها به حوزه صوت نگاشت می‌شوند و سپس ویژگی‌های مورد نیاز از جمله طیف‌های مورد نظر استخراج می‌شوند. این طیف‌های دوبعدی همانند تصاویر عمل می‌کنند. این تصاویر با استفاده از تنظیم مجدد مدل‌های از پیش آموزش دیده و با کمک یادگیری انتقالی، طبقه‌بندی می‌شوند. این روش به دلیل نیاز نداشتن به مهندسی معکوس برنامه‌ها و داشتن دیدگاه جعبه سیاه‌گونه، دقت و کارایی بالایی در تشخیص بدافزارها ارائه می‌دهد. راه‌کار پیشنهادی شامل چهار مرحله اصلی است: (۱) آماده‌سازی داده‌ها، (۲) انجام تحلیل ایستا، (۳) طراحی، آموزش

و آمایش مدل و (۴) ارزیابی نهایی. در کل، این روش با استفاده از داده‌های صوتی در تحلیل ایستا، دقت تشخیص را افزایش داده و چالش‌های ناشی از روش‌های مبهم‌سازی در فرایند مهندسی معکوس را کاهش می‌دهد.

۱-۳- فاز نخست: جمع‌آوری داده‌ها

فاز نخست مربوط به جمع‌آوری داده‌ها است. یک رویکرد مبتنی بر یادگیری عمیق، در وهله نخست نیاز به حجم زیادی از داده‌های ورودی دارد. مدلی با معماری شبکه عصبی عمیق که شامل میلیون‌ها مؤلفه برای آموزش است؛ در صورتی می‌تواند به خوبی آموزش ببیند که با حجم زیادی از داده‌های مناسب، تغذیه شود؛ به همین دلیل در فاز نخست، نیاز به جمع‌آوری مجموعه‌ای بزرگ و به اندازه کافی متنوع از داده‌ها است. نمودار شکل (۳)، فرایند جمع‌آوری داده‌های مورد نیاز را نشان می‌دهد.



(شکل-۲): نمودار فعالیت فاز نخست، جمع‌آوری داده‌ها
(Figure-3): Activity diagram for the first phase, data collection

حدود ۱۳۰ گیگابایت حجم دارد که جزئیات تعداد برنامه‌ها در جدول (۱) ارائه شده است.

(جدول-۱): تعداد داده‌ها در بخش تحلیل ایستا

(Table-1): Number of data in the static analysis section

تعداد برنامه‌ها	
۵۶۸۷	خوش خیم
۵۶۸۷	بدخیم
۱۱۳۷۴	مجموع

از آنجا که یکی از اهداف مدنظر ما در این پژوهش، بررسی عملکرد نهایی مدل بر روی برنامه‌های مبهم‌شده بود، این

در این پژوهش، داده‌های مورد نیاز شامل فایل‌های APK برنامه‌های اندرویدی است که برای افزایش دقت و جامعیت داده‌ها، از مجموعه‌داده‌های معتبر CIC استفاده شد. این مجموعه‌داده‌ها که بیش از دویست گیگابایت حجم دارند و شامل بیش از نوزده هزار برنامه است، اطلاعات ایستا و پویای کاملی از بدافزارهای اندرویدی با تنوع بالا ارائه می‌دهند. دسترسی آسان و قابلیت مقایسه نتایج پژوهش با سایر مطالعات مشابه از مزایای این مجموعه‌داده‌ها محسوب می‌شود. این مجموعه‌داده، پس از حذف فایل‌های ناقص

۲-۳- فاز دوم: تحلیل ایستای برنامه‌ها

این فاز شامل انجام فرایندهای تحلیلی بر روی برنامه‌های جمع‌آوری شده در فاز نخست است. در این فاز برای استخراج ویژگی‌ها و اطلاعات مورد نیاز از هر برنامه، از تحلیل ایستا استفاده شد. شکل (۴)، نمودار فعالیت تحلیل ایستای برنامه‌ها را نشان می‌دهد.

این فاز شامل انجام تحلیل ایستای برنامه‌ها است؛ بنابراین پژوهش‌های بررسی شده، استفاده از اطلاعات موجود در بایت‌کدهای خام فایل‌هایی مانند فایل APK و Classes.dex، بسیار مؤثر خواهد بود. بایت‌کدهای یک فایل اجرایی در واقع یک نوع بازنمایی از رفتار برنامه است [۲۴، ۲۵]؛ بنابراین با تحلیل درست بایت‌کدهای فایل اجرایی برنامه‌ها، می‌توان دید خوبی از خوش‌خیم بودن یا نبودن آن به دست آورد.

در پژوهش‌هایی مانند [۱۳، ۱۴، ۲۴]، با استفاده از یک نگاشت بین مقدار بایت‌کدها و پیکسل‌های تصویر، یک بازنمایی تصویری از فایل‌های اجرایی به دست می‌آورند؛ از آنجا که بایت‌کدهای یک فایل اجرایی به صورت پشت سرهم معنا دارند، تبدیل فهرستی از داده‌های پشت سرهم به ساختار داده‌ای مانند تصویر که دارای سطرها و ستون‌های مختلف است، باعث می‌شود اطلاعات نهفته در توالی این داده‌ها به هم ریخته و در گام‌های بعدی، خطاهای احتمالی را به وجود آورد.

یک ساختار داده بسیار عالی و قابل تحلیل که از پشت سرهم قرارگرفتن داده‌های مختلف ایجاد می‌شود، فایل‌های صوتی است. سیگنال‌های صوتی با داشتن داده‌های مختلف در زمان‌های مختلف، می‌توانند بهترین بازنمایی برای بایت‌کدهای خام باشند؛ بنابراین در این مرحله نیاز است تا برنامه‌ها به فایل‌های صوتی مختلف، نگاشت شوند.

برای انجام این تبدیل، یک فرایند نگاشت سه مرحله‌ای در نظر گرفته شده است که در شکل (۵) نمودار فعالیت آن ارائه شده است.

مجموعه داده از جهت دیگر که وجود برنامه‌های مبهم‌شده واقعی در هر دو دسته برنامه‌های خوش‌خیم و بدخیم است، برای ما اهمیت دارد. با کمک ابزار ApkId که برای اهداف مختلفی در تحلیل فایل‌های APK استفاده می‌شود، مجموعه داده حاضر مورد بررسی قرار گرفت. در جدول (۲)، تعداد برنامه‌های مبهم‌شده موجود در مجموعه داده مورد استفاده ارائه شده است.

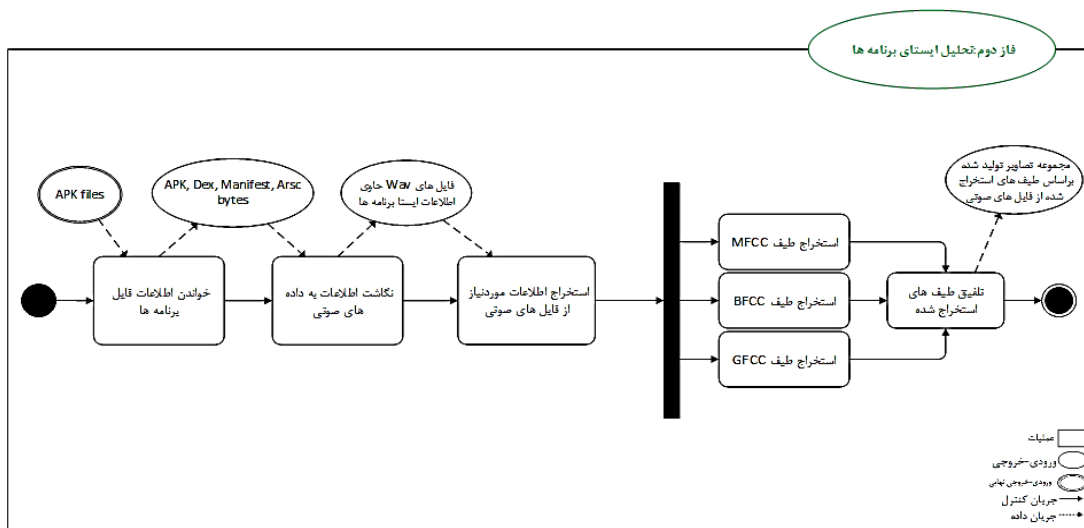
(جدول ۲): تعداد برنامه‌های مبهم‌شده

(Table-2): Number of obfuscated applications

تعداد برنامه‌های مبهم	
۳۲۲	خوش‌خیم
۵۷۰	بدخیم
۸۹۲	مجموع

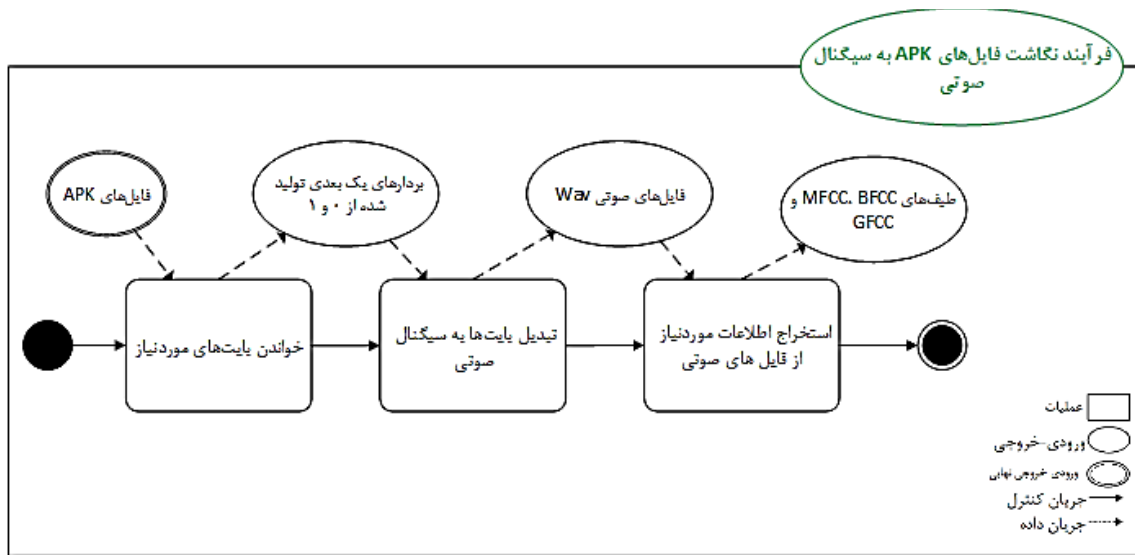
با توجه به جدول (۲)، حدود ۷.۸ درصد از کل مجموعه داده‌ها را برنامه‌های مبهم‌شده شامل می‌شود که این مورد بسیار مناسبی برای استفاده در پژوهش جاری است. نگاه جعبه‌سیاه‌گونه روش مورد استفاده در راه کار پیشنهادی ما که توضیحات آن در بخش‌های آینده ارائه شده است، در تشخیص برنامه‌های مبهم‌شده نیز بسیار مناسب عمل کرده است؛ از طرف دیگر، وجود این تعداد برنامه واقعی که از سناریوهای واقعی مبهم‌سازی استفاده می‌کنند، می‌تواند نتایج ارزیابی راه کار پیشنهادی را بسیار ارزشمندتر کند؛ چراکه با کمک داده‌های مبهم‌شده واقعی و نه فرایندهای تغییر بسته و یا مبهم‌سازی مجدد، انجام شده است.

مجموعه داده مورد استفاده در این پژوهش، بر اساس روش ارزیابی بیرون نگهدارنده، به سه قسمت برای آموزش، ارزیابی و آزمون در نظر گرفته می‌شود. برای این منظور، هشتاد درصد از داده‌ها به عنوان داده آموزشی، ده درصد از داده‌ها به عنوان داده ارزیابی و ده درصد نهایی نیز به عنوان داده آزمون در نظر گرفته می‌شود. داده‌های آموزشی در فرایند آموزش به مدل وارد می‌شوند. از داده‌های ارزیابی در طی فرایند آموزش و برای تنظیم بهینه وزن‌ها و مؤلفه‌های مدل استفاده می‌شود. از داده‌های آزمون نیز برای ارزیابی نهایی مدل آموزش دیده استفاده می‌شود.



(شکل ۳): نمودار فعالیت فاز دوم از راه کار پیشنهادی

(Figure-4): Activity diagram for the second phase



(شکل-۴): نمودار فعالیت فرایند نگاشت فایل‌های APK به سیگنال‌های صوتی
(Figure-5): Activity diagram of the process of mapping APK files to audio signals

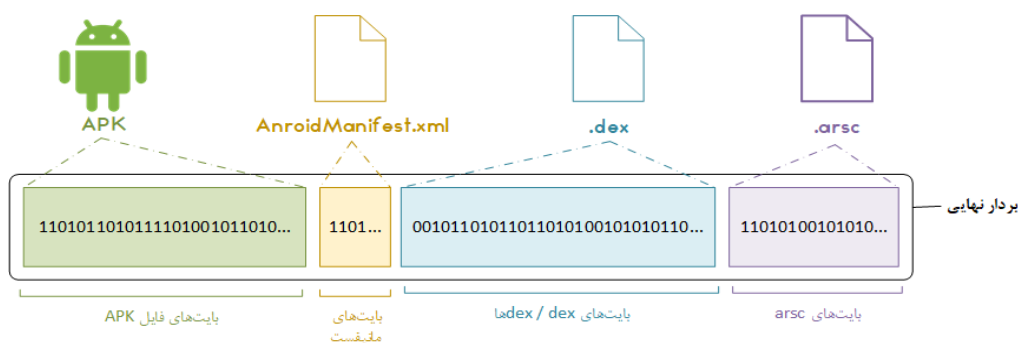
بردار عددی به‌دست‌آمده از مرحله پیشین باید به یک سیگنال صوتی مناسب تبدیل شود که برای این منظور ابتدا مشخصات سیگنال تعیین می‌شود؛ از آنجا که سیگنال صوتی نهایی تنها برای آموزش مدل یادگیری عمیق استفاده می‌شود و نیازی به شنونده انسانی ندارد، افزایش نرخ داده آن تنها باعث افزایش حجم داده‌ها خواهد شد [۲۵]؛ همچنین، استفاده از یک کانال صوتی به‌جای دو کانال به کاهش حجم کمک می‌کند. نرخ نمونه‌برداری سیگنال، که تعداد نمونه‌ها در هر ثانیه را مشخص می‌کند، در این پژوهش شانزده کیلوهرتز در نظر گرفته شده است [۲۵] که علاوه بر حفظ داده‌های اصلی، نیاز به پهنای باند کمتری دارد. عمق بیت سیگنال نیز شانزده بیت تعیین شده است که امکان پشتیبانی از ۶۵۵۳۶ مقدار مختلف را فراهم کرده و بازنمایی دقیق‌تری ارائه می‌دهد؛ درنهایت، سیگنال به‌صورت غیرفشرده و خام با فرمت WAV و نوع uPCM ذخیره می‌شود؛ زیرا استفاده از فشرده‌سازی‌هایی مانند MP3 ممکن است داده‌های مرحله پیشین را تغییر دهد که مطلوب ما نیست.

با توجه به شکل (۵)، فرایند سه مرحله‌ای نگاشت با خواندن بایت‌های مورد نیاز از فایل‌ها آغاز می‌شود. در ادامه بایت‌های خوانده‌شده با یک سازوکار خاص به سیگنال‌های صوتی تبدیل شده و درانتها، اطلاعات مورد نیاز از این سیگنال‌های صوتی استخراج می‌شود. در ادامه توضیحات این مراحل ارائه شده است.

مرحله نخست: استخراج بایت‌های فایل APK

فایل‌های APK که به‌عنوان فایل‌های فشرده حاوی کدهای اجرایی، فایل‌های Manifest و منابع دیگر هستند، از حالت فشرده خارج شده و بایت‌های مربوط به فایل Classes.dex، AndoridManifest.xml و resources.arsc پشت سرهم قرار می‌گیرند. خروجی این مرحله یک بردار یک‌بعدی از بایت‌ها با مقادیر عددی بین ۰ تا ۲۵۵ است که هر بایت با هشت بیت نمایش داده می‌شود که در شکل (۶)، شمای کلی از این فرایند ارائه شده است.

مرحله دوم: تبدیل بایت‌ها به سیگنال صوتی



(شکل-۵): کنار هم قرار گرفتن بایت‌های فایل‌های APK، Manifest، dex و arsc و تولید بردار نهایی
(Figure-6): Putting together the bytes of the APK, Manifest, dex, and arsc files and generating the final vector

سیگنال صوتی نهایی با نرخ نمونه‌برداری شانزده کیلوهرتز، عمق بیت برابر با شانزده بیت و نوع فشرده‌سازی PCM خام در قالب صوت تک‌کاناله و با فرمت WAV ایجاد خواهد شد؛ بنابراین در این مرحله، هر شانزده بیت از بردار ورودی به این مرحله،

سیگنال صوتی نهایی با نرخ نمونه‌برداری شانزده کیلوهرتز، عمق بیت برابر با شانزده بیت و نوع فشرده‌سازی PCM خام در قالب

به عنوان یک نمونه صوتی در نظر گرفته می شود و هر شانزده هزار نمونه در یک ثانیه از سیگنال صوتی نهایی قرار خواهند گرفت. این فرایند با کمک بسته نرم افزاری رسمی Wav در Python انجام می شود.

مرحله سوم: استخراج ویژگی های سیگنال صوتی

سیگنال های صوتی به دست آمده برای استخراج ویژگی های دوبعدی مانند ضرایب Cepstral پردازش می شوند. این ویژگی ها، سیگنال را از حوزه زمان-دامنه به حوزه زمان-فرکانس منتقل می کنند و به عنوان تصاویر دوبعدی برای استفاده در مدل های یادگیری عمیق ذخیره می شوند. این تصاویر مبنای تحلیل در مراحل بعدی پژوهش اند.

در این پژوهش، سه طیف اصلی از ضرایب صوتی شامل MFCC¹، BFCC² و GFCC³ برای استخراج ویژگی های دوبعدی سیگنال صوتی مورد استفاده قرار گرفته اند. ضرایب MFCC بر اساس دستگاه شنوایی و ادراک صوتی انسان طراحی شده اند [۲۶] و به دلیل تطابق با ویژگی های شنیداری، کاربرد گسترده ای در تشخیص و شناسایی گفتار دارند. ضرایب BFCC که مشابه MFCC استخراج می شوند، از بانک پالایه های Bark به جای پالایه های Mel استفاده می کنند. این ضرایب به تازگی در تشخیص بدافزارهای اندروید به کار گرفته شده اند و توانایی بالایی در تمایز بدافزارها نشان داده اند [۲۵، ۲۷]. ضرایب GFCC نیز برای شبیه سازی دقیق تر پردازش صوتی حلزون گوش انسان طراحی شده اند و با توجه به شباهت زیاد به نحوه پردازش صوتی انسان، دقت و مقاومت بیشتری در برابر نوفه دارند. این ضرایب نیز به تازگی در شناسایی بدافزارها استفاده شده و عملکرد قابل توجهی از خود نشان داده اند [۲۵، ۲۸].

تا این مرحله، به ازای هر فایل برنامه در مجموعه داده ها، سه بردار دوبعدی MFCC، BFCC و GFCC که از سیگنال صوتی استخراج شده اند، در اختیار داریم. ابعاد این بردارها با یکدیگر برابر بوده و هر کدام را می توان به عنوان یک تصویر تک رنگ در نظر گرفت. با توجه به عملکرد بهتر مدل های یادگیری عمیق در طبقه بندی تصاویر رنگی به دلیل وجود اطلاعات بیشتر در این تصاویر نسبت به تصاویر سیاه و سفید [۱۷]، این سه بردار برای ایجاد یک تصویر رنگی سه کاناله مورد استفاده قرار می گیرند؛ بدین ترتیب، تصویر MFCC در کانال قرمز (Red)، تصویر BFCC در کانال سبز (Green) و تصویر GFCC در کانال آبی (Blue) قرار داده می شود؛ نتیجه نهایی، یک تصویر رنگی سه کاناله است که در شکل (۷) به صورت گرافیکی نمایش داده شده است.

با استفاده از این روش، علاوه بر ایجاد یک تصویر رنگی و تلفیق اطلاعات سه ویژگی مختلف در یک داده غنی شده، نیاز به آموزش سه مدل مختلف برای طبقه بندی هر کدام از این ویژگی ها حذف شد؛ در نتیجه، به ازای هر برنامه در مجموعه داده، یک تصویر رنگی با

برچسب برنامه خوش خیم یا بدخیم وجود دارد. این فرایند با تحلیل ایستای برنامه ها آغاز شد و با توجه به شباهت توالی بایت ها در برنامه ها به توالی اطلاعات در سیگنال های صوتی، یک فرایند نگاشتی طراحی شد که با نگاهی جعبه سیاه گونه و حذف پیچیدگی های متداول در تجزیه بسته های برنامه، هر برنامه را به یک سیگنال صوتی تبدیل کرد؛ سپس طیف های MFCC، BFCC و GFCC از این سیگنال ها استخراج شد و با تلفیق این سه طیف، به ازای هر برنامه، یک تصویر رنگی ایجاد شد. در ادامه، از ظرفیت بالای مدل های یادگیری عمیق در حوزه پردازش و طبقه بندی تصویر برای انجام فرایند تشخیص استفاده خواهد شد.

۳-۳- فاز سوم: طراحی، آموزش و آزمایش مدل ها

پس از جمع آوری داده ها در فاز نخست و انجام تحلیل های مورد نیاز در فاز دوم، مجموعه ای از اطلاعات مربوط به داده های بخش تحلیل ایستا وجود دارد. سناریوهای مختلفی برای طراحی و آموزش یک مدل یادگیری عمیق وجود دارد از جمله طراحی و آموزش یک مدل یادگیری عمیق از ابتدا و یا استفاده از مدل های از پیش آموزش دیده که باید برای تنظیم مجدد آن ها اقدام کرد؛ از آنجا که پژوهش های مختلف [۱۴، ۱۶، ۲۰] نتایج بسیار مناسبی با استفاده از مدل های از پیش آموزش دیده به دست آورده اند، پژوهش گران مقاله حاضر نیز تصمیم بر استفاده از این نوع از مدل ها و سپس تنظیم مجدد کردن آن گرفتند؛ همان طور که در بخش پیشین مشاهده شد، به ازای هر برنامه اندرویدی موجود در مجموعه داده اولیه، یک تصویر رنگی در اختیار است که از ترکیب سه ویژگی MFCC، BFCC و GFCC به وجود آمده است. این سه طیف ویژگی بر مبنای سیگنال صوتی تولیدی از بایت کدهای برنامه به وجود آمده اند؛ بنابراین در این قسمت پژوهش گران با یک مسئله طبقه بندی تصاویر رنگی مواجه اند. طبقه بندی تصاویر از مسائل بسیار معروف حوزه پردازش تصویر است. در این مسائل، یک مدل یادگیری عمیق با کمک مجموعه ای از داده های تصویری، آموزش دیده و سپس طبقه بندی تصاویر را بر اساس برچسب های ورودی انجام می دهد. مدل های یادگیری عمیق زیادی وجود دارند که با معماری های بهینه، روی حجم بزرگی از داده های تصویری آموزش دیده اند تا بتوانند فرایند طبقه بندی تصاویر را با دقت بسیار بالایی انجام دهند [۲۹-۳۲].

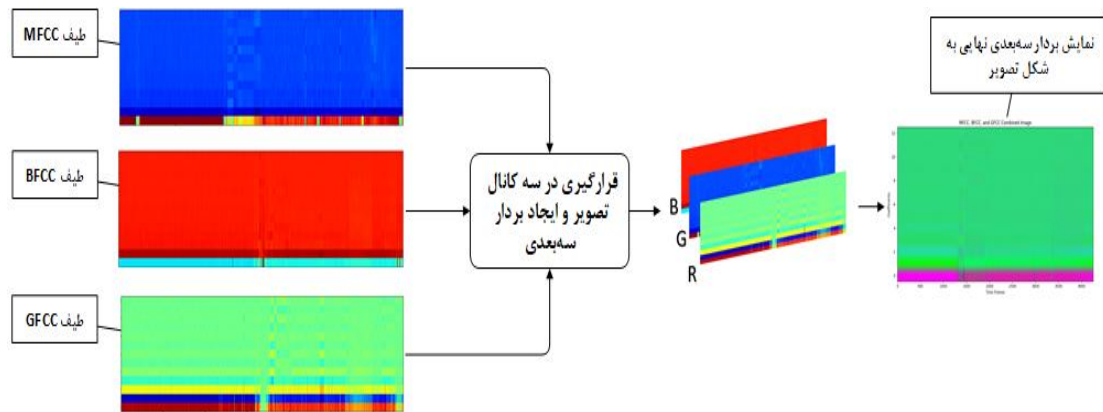
از مدل های معروف می توان به MobileNetV2 با حجم چهارده مگابایت و ۳.۵ میلیون مؤلفه اشاره کرد که یکی از سبک ترین مدل های سبک است که روی مجموعه داده ImageNet شامل ۱.۲ میلیون تصویر آموزش دیده و می تواند هزار کلاس مختلف را تشخیص دهد؛ در مقابل، مدل های پیچیده تری مانند VGG16 و NASNetLarge با حجم های بزرگ تر (به ترتیب ۵۲۸ و ۳۴۳ مگابایت) و تعداد مؤلفه های بیشتر (۱۳۸.۴ و ۸۸.۹ میلیون) طراحی شده اند که نشان دهنده توانایی بالاتر آن ها در پردازش داده های پیچیده است. سایر مدل ها مانند Xception، InceptionV3 و EfficientNetB7 نیز با حجم ها و تعداد

¹ Mel Frequency Cepstral Coefficients

² Bark Frequency Cepstral Coefficients

³ Gammatone Frequency Cepstral Coefficients





و ایجاد یک بردار سه بعدی MFCC، BFCC و GFCC (شکل-۶): ترکیب سه طیف
(Figure-7): Combining the three spectra MFCC, BFCC, and GFCC and creating a 3D vector vector

تلفن همراهها و سامانه‌های نهفته. این ویژگی‌ها آن را گزینه‌ای ایدئال برای پژوهش حاضر می‌کند.

در این پژوهش، مدل EfficientNetV2B0 به عنوان مدل پایه انتخاب شد. تصاویر ورودی به ابعاد (۳، ۲۲۴، ۲۲۴) تغییر اندازه داده شدند. بخش انتهایی مدل پایه با شبکه تمام متصل جایگزین شد تا تصاویر برنامه‌های خوش‌خیم و بدخیم را طبقه‌بندی کند. لایه نهایی شامل یک نرون با تابع فعال‌سازی Sigmoid است که خروجی صفر یا یک را ارائه می‌دهد.

برای جلوگیری از بیش‌برازش^۱ مدل، از لایه Dropout استفاده شد. این لایه به طور تصادفی برخی از نرون‌ها را در زمان آموزش غیرفعال می‌کند. نرخ Dropout بین ۰.۲ تا ۰.۵ تنظیم شد تا از وابستگی بیش‌ازحد مدل به داده‌های خاص جلوگیری شود. تابع فعال‌سازی ReLU در لایه‌های Dense انتخاب شد؛ زیرا علاوه بر کارایی بالا، ویژگی‌های غیرخطی پیچیده‌تری را نیز می‌آموزد.

برای اتصال معماری جدید به مدل پایه، از لایه GlobalAveragePooling2D استفاده شد. این لایه تعداد مؤلفه‌ها را کاهش داده و از آموزش بیش‌برازش جلوگیری می‌کند؛ همچنین، اطلاعات توزیعی در نقشه ویژگی‌ها را حفظ می‌کند و دقت نهایی مدل را افزایش می‌دهد.

برای آموزش مدل، از بهینه‌ساز Adam استفاده شد که نرخ یادگیری متغیر و سرعت هم‌گرایی بالایی دارد. این بهینه‌ساز برای وظایف طبقه‌بندی دودویی به دلیل توانایی مقابله با داده‌های پیچیده و نوفه‌ها، بسیار مناسب است. تابع هزینه انتخاب شده، آنتروپی متقابل دودویی بود که به طور مؤثر اختلاف بین پیش‌بینی و برچسب واقعی را محاسبه می‌کند.

مدل‌هایی که بر روی مجموعه داده‌های بزرگ و عمومی مانند ImageNet آموزش دیده‌اند، توانایی بالایی در درک بصری دارند. این مدل‌ها می‌توانند ویژگی‌های مؤثر بصری را از تصاویر استخراج و با استفاده از یادگیری انتقالی، بدون نیاز به آموزش از ابتدا، در مجموعه داده‌های کوچک‌تر یا خاص‌تر نیز به خوبی عمل کنند. این رویکرد باعث تسریع و افزایش دقت فرایند یادگیری می‌شود.

EfficientNetV2، خانواده‌ای از مدل‌های شبکه عصبی پیچشی است که توسط پژوهش‌گران Google به عنوان نسخه بهبودیافته مدل EfficientNet معرفی شده است. این مدل‌ها که برای طبقه‌بندی تصاویر طراحی شده‌اند، از روش آموزش تدریجی استفاده می‌کنند؛ در این روش، مدل با تصاویر ساده‌تر شروع به یادگیری می‌کند و سپس با تصاویر پیچیده‌تر ادامه می‌دهد [۳۲]. این استراتژی نه تنها زمان آموزش را کاهش می‌دهد، بلکه مصرف منابع محاسباتی را نیز بهینه می‌کند.

خانواده EfficientNetV2 شامل مدل‌هایی با اندازه‌ها و پیچیدگی‌های مختلف مانند EfficientNetV2-S، EfficientNetV2-M و EfficientNetV2-L است. این مدل‌ها برای کاربردهایی با محدودیت منابع یا وظایف پیچیده‌تر طراحی شده‌اند. EfficientNetV2-S برای محیط‌هایی با محدودیت منابع مناسب است؛ در حالی که EfficientNetV2-L برای دقت بالاتر در وظایف پیچیده‌تر استفاده می‌شود.

مدل EfficientNetV2 از چندین مزیت کلیدی برخوردار است: (۱) زمان آموزش کمتر و مصرف حافظه پایین‌تر به دلیل یادگیری تدریجی، (۲) دقت بالاتر در طبقه‌بندی تصاویر نسبت به مدل‌های قدیمی‌تر مانند ResNet و MobileNet، (۳) قابلیت پیاده‌سازی بهینه در دستگاه‌های با محدودیت سخت‌افزاری مانند

^۱ Overfit

(جدول-۳): معماری و مشخصات مدل

(Table-۳): Model architecture and specifications

نام لایه	تعداد کل مؤلفه‌ها	ابعاد خروجی	تابع فعال ساز	توضیحات
مدل پایه EfficientNetV2B0	حدود ۶ میلیون	—	—	—
GlobalAveragePooling2D	—	اندازه خروجی: ۱۲۸۰	—	—
Dense	۵۱۲	اندازه خروجی: ۵۱۲	ReLU	—
Dropout	۵۱۲	اندازه خروجی: ۵۱۲	—	نرخ: ۰.۵
Dense	۲۵۶	اندازه خروجی: ۲۵۶	ReLU	—
Dropout	۲۵۶	اندازه خروجی: ۲۵۶	—	نرخ: ۰.۵
Dense	۱	اندازه خروجی: ۱	Sigmoid	—

(جدول-۴): تعداد مؤلفه‌های مدل نهایی در حالات مختلف

(Table-۴): Number of parameters of the final model in different cases

مشخصات مدل در زمان قابل آموزش بودن مدل پایه		مشخصات مدل در زمان قابل آموزش بودن مدل پایه	
تعداد کل مؤلفه‌های مدل	۶۷۰۶۷۶۹	تعداد کل مؤلفه‌های مدل	۶۷۰۶۷۶۹
مؤلفه‌های قابل آموزش	۶۶۴۶۱۶۱	مؤلفه‌های قابل آموزش	۶۶۴۶۱۶۱
مؤلفه‌های غیرقابل آموزش	۶۰۶۰۸	مؤلفه‌های غیرقابل آموزش	۶۰۶۰۸

مدل را در وظایف طبقه‌بندی دودویی بهبود می‌بخشد، بلکه تعمیم‌پذیری آن را نیز افزایش می‌دهد؛ همچنین، با مدیریت تدریجی داده‌ها و کاهش وابستگی سریع مدل به داده‌های جدید، مشکل آموزش بیش‌ازحد کاهش قابل توجهی می‌یابد. چالش مدیریت حجم بزرگ داده‌ها به کمک طراحی یک خط لوله داده‌ای حل شد که با استفاده از TensorFlow پیاده‌سازی شده‌است؛ این خط لوله، داده‌ها را به‌صورت پویا در دسته‌های کوچک بارگذاری کرده و مراحل پردازش مانند استخراج طیف‌ها را همزمان و در هنگام بارگذاری انجام می‌دهد. با این روش، مشکل محدودیت حافظه در حین آموزش مدل برطرف و فرایند آموزش بهینه‌تر شده‌است.

تمامی مراحل پیاده‌سازی و آموزش مدل در ویندوز یازده و با زبان پایتون نسخه ۳.۱۱.۱۰ انجام شده‌است؛ همچنین از کتابخانه‌های Tensorflow نسخه ۲.۱۷.۰ و NumPy نسخه ۱.۲۶.۴ برای آموزش مدل، پردازش داده‌ها و تحلیل نتایج استفاده شده‌است؛ همچنین فرایند آموزش مدل با اندازه دسته برابر با هشت و نرخ یادگیری برابر با ۰.۰۰۱ انجام شده‌است.

در این پژوهش، مدل جدید مبتنی بر یادگیری انتقالی طراحی و آموزش داده شده‌است. با مدیریت کارآمد داده‌ها و استفاده از معماری پیشرفته EfficientNet، پژوهش‌گران موفق شدند بر محدودیت‌های موجود غلبه کرده و عملکرد مدل را در طبقه‌بندی تصاویر بهبود دهند. در بخش‌های بعدی، نتایج و ارزیابی عملکرد مدل به‌صورت کامل ارائه شده‌است.

۳- فاز چهارم: ارزیابی

برای ارزیابی عملکرد مدل آموزش‌دیده در شناسایی بدافزارهای اندرویدی، از معیارهای مختلفی استفاده می‌شود تا دقت و کارایی آن مشخص شود. صحت نشان‌دهنده درصد کلی نمونه‌هایی است که به‌درستی طبقه‌بندی شده‌اند. فراخوانی به کارایی مدل در تشخیص نمونه‌های بدافزار واقعی اشاره دارد؛ درحالی‌که دقت

همان‌طور که از جدول (۳) نیز مشخص است، در مدل شبکه عصبی عمیق این پژوهش، ابتدا مدل پایه قرار داشته و سپس لایه‌های جدید قرار گرفته‌اند. جدول (۴)، جزئیات مدل مثل تعداد مجموع مؤلفه‌های آن و تعداد مؤلفه‌های غیرقابل آموزش آن را نشان می‌دهد.

فرایند یادگیری انتقالی^۱ یکی از روش‌های پرکاربرد و کارآمد در یادگیری عمیق است که به‌جای آموزش مدل از ابتدا، از دانش مدل‌های از پیش آموزش‌دیده برای وظایف جدید استفاده می‌کند [۱۶، ۱۴]. این روش به‌ویژه در مواردی که داده‌های آموزشی محدودند، مؤثر است. مدل پایه که بر روی مجموعه داده‌های بزرگ و متنوع آموزش‌دیده، به‌عنوان منبع استخراج ویژگی‌های عمومی استفاده شده‌است و سپس این ویژگی‌ها برای آموزش مدل جدید و خاص‌منظوره به کار می‌روند؛ این روش علاوه بر کاهش زمان و منابع محاسباتی، دقت و کارایی مدل را بهبود می‌بخشد [۱۹].

در پژوهش حاضر، فرایند یادگیری انتقالی با استفاده از معماری پایه EfficientNetV2B0 و افزودن لایه‌های جدید تمام‌متصل انجام شده‌است. استراتژی آموزش به‌صورت تدریجی و با فعال و غیرفعال کردن قابلیت یادگیری مدل پایه طراحی شده‌است تا از بیش‌برازش جلوگیری شود. در ابتدا، مدل پایه غیرقابل آموزش‌شده و لایه‌های جدید بر روی ویژگی‌های استخراج‌شده تنظیم می‌شوند و آموزش می‌بینند؛ سپس، با فعال‌سازی مجدد مدل پایه، ویژگی‌های آن برای تطبیق با داده‌های خاص بهینه‌سازی می‌شوند. این روش تعادلی میان استفاده از دانش عمومی و بهینه‌سازی تخصصی‌تر مدل برای وظیفه جدید، ایجاد می‌کند.

این چرخه آموزش تدریجی، موجب می‌شود، مدل از ویژگی‌های کلیدی مدل پایه بهره‌برد و درعین حال به تطبیق بهتر با داده‌های جدید دست یابد. این رویکرد نه تنها عملکرد

¹ Transfer Learning

برای تحلیل عملکرد مدل در تشخیص نمونه‌های مثبت و منفی، از ماتریس اغتشاش استفاده می‌شود که تعداد نمونه‌های مثبت و منفی حقیقی و کاذب را مشخص می‌کند؛ همچنین، برای ارزیابی جامع‌تر، از منحنی ROC و معیار AUC بهره گرفته می‌شود؛ منحنی ROC نرخ تشخیص مثبت حقیقی در برابر مثبت کاذب را نمایش می‌دهد و مقدار AUC سطح زیر این منحنی را اندازه‌گیری می‌کند. مقدار بالاتر AUC نشان‌دهنده کارایی بهتر مدل و استقلال آن از آستانه تصمیم‌گیری است. این معیارها با نتایج پژوهش‌های معتبر مقایسه شده و نقاط قوت و ضعف مدل در ادامه تحلیل خواهند شد.

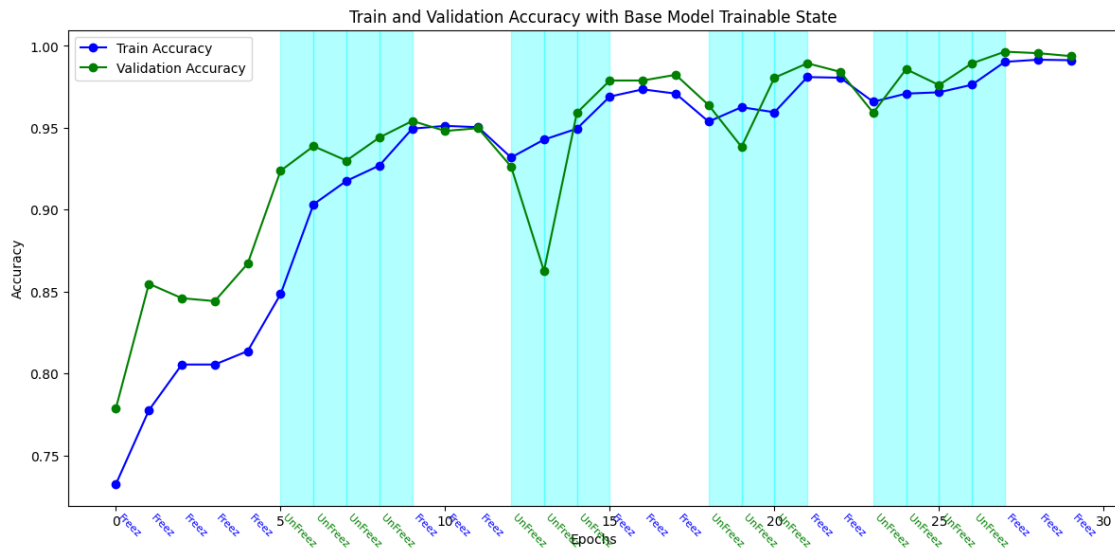
میزان صحت تشخیص‌های بدافزار را ارزیابی می‌کند. معیار F1 با محاسبه میانگین هارمونیک دقت و فراخوانی، تعادل بین این دو را نشان می‌دهد. روابط (۱ تا ۴)، نحوه محاسبه این معیارها را نشان می‌دهد.

$$(1) \text{ Accuracy} = (TP+TN)/(TP+FP+TN+FN)$$

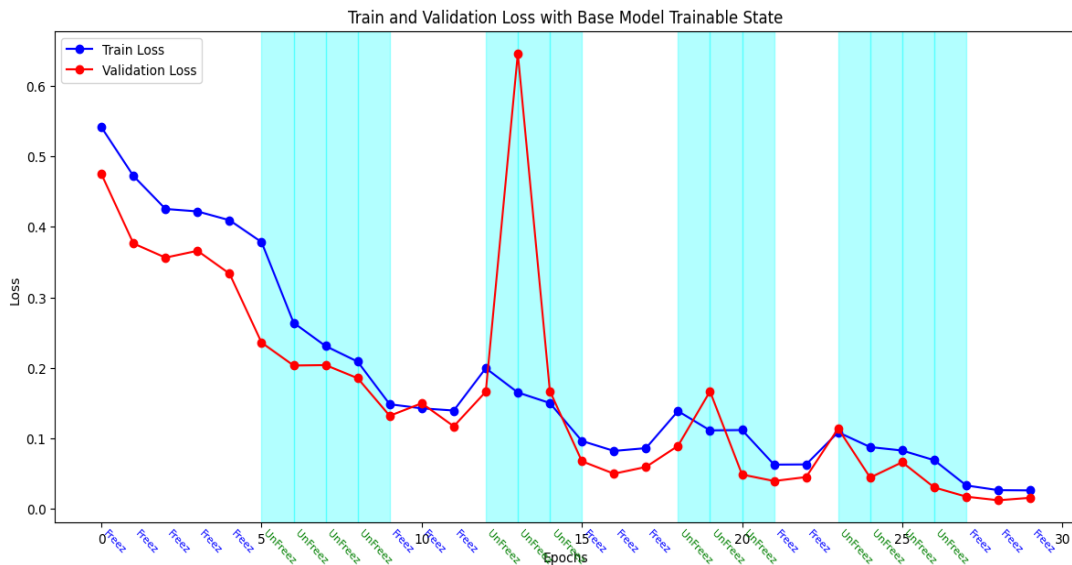
$$(2) \text{ Recall} = TP/(TP+FN)$$

$$(3) \text{ Precision} = TP/(TP+FP)$$

$$\text{F-measure} = (2 \times \text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$$



(شکل-۷): نمودار صحت مدل در train و validation. قابل آموزش بودن یا نبودن مدل پایه با زمینه آبی رنگ در نمودار مشخص شده است (Figure-8): Model accuracy chart in train and validation. The freezing state of the base model is indicated by the blue background in the chart.



(شکل-۸): نمودار هزینه مدل در train و validation. قابل آموزش بودن یا نبودن مدل پایه با زمینه آبی رنگ در نمودار مشخص شده است (Figure-9): Model loss chart in train and validation. The freezing state of the base model is indicated by the blue background in the chart.

مشاهده می‌شود، با پیشرفت آموزش، صحت مدل در مجموعه اعتبارسنجی افزایش یافته که نشان‌دهنده کارایی مطلوب فرایند آموزش است.

در شکل (۸)، نمودار صحت مدل در طول فرایند آموزش نمایش داده شده است. در بخش‌هایی با زمینه سفید، وزن‌های مدل پایه غیرقابل آموزش بوده و در بخش‌هایی با زمینه آبی، وزن‌های مدل پایه در حال بهینه‌سازی هستند؛ همان‌طور که

به وسیله مدل دیده نشده بودند، با دقت بالایی طبقه بندی شده اند. ماتریس نشان می دهد که مدل ۱۱۳۸ نمونه را به درستی به عنوان برنامه های خوش خیم و ۱۱۲۲ نمونه را به درستی به عنوان بدافزار شناسایی کرده است. خطاهای مدل بسیار محدود بوده و به ترتیب تنها چهار نمونه خوش خیم و هشت نمونه بدخیم به اشتباه برچسب خورده اند. این دقت بالا بیان کننده توانایی چشم گیر مدل در تفکیک داده هاست؛ هر چند باید به تهدید بالقوه ناشی از شناسایی اشتباه هشت بدافزار به عنوان برنامه خوش خیم توجه داشت. با افزایش حجم و تنوع داده ها در آموزش های بعدی می توان این خطاها را کاهش داد.

تحلیل شاخص های ارزیابی نیز این نتایج را تقویت می کند. صحت ۹۹.۳ درصد نشان دهنده توانایی کلی مدل در تشخیص درست داده ها است. دقت ۹۹.۸ درصد تأیید می کند که نمونه های مثبت طبقه بندی شده بیشتر متعلق به برچسب صحیح بوده اند و فراخوانی ۹۹.۱ درصد بیان کننده آن است که بخش عمده ای از برنامه های بدخیم به درستی شناسایی شده اند. مقدار بالای F-measure تعادل میان دقت و فراخوانی را نشان می دهد. علاوه بر این، مدل توانسته است تمامی برنامه های مبهم شده موجود در داده های آزمون را بدون هیچ خطایی شناسایی کند و نرخ تشخیص برنامه های مبهم شده روش پیشنهادی به صد درصد در این مجموعه داده به ثبت برسد. این ترکیب از شاخص ها و نتایج، بیان کننده طراحی و آموزش دقیق مدل با کارایی بالا و خطای اندک است.

نتایج به دست آمده از مدل پیشنهادی، نه تنها در معیارهای استاندارد ارزیابی مانند دقت، صحت و فراخوانی عملکرد بسیار خوبی از خود نشان داده است، بلکه در مواجهه با یکی از چالش برانگیزترین مسائل حوزه تشخیص بدافزار یعنی شناسایی برنامه های مبهم شده نیز به موفقیت دست یافته است. تمام نمونه های مبهم شده موجود در مجموعه داده آزمون، بدون خطا به وسیله مدل شناسایی شدند. این موضوع، یکی از برجسته ترین مزایای رویکرد پیشنهادی به شمار می رود؛ زیرا بسیاری از مدل های سنتی یا حتی یادگیری عمیق در مواجهه با این گونه نمونه ها، دچار خطا یا کاهش دقت می شوند.

همچنین، برای تحلیل بهتر عملکرد مدل در مواجهه با داده های نادیده، از منحنی ROC و مقدار AUC استفاده شد. مقدار AUC معادل ۰.۹۹۳۸ بر روی داده های آزمون و تفاوت ناچیز آن با داده های آموزش، نشان می دهد که مدل دچار بیش برآزش نشده و دارای توان تعمیم بالا است؛ بنابراین با وجود دقت بالا، ساختار آموزش تدریجی و استفاده از یادگیری انتقالی، از بروز بیش برآزش نیز جلوگیری شده است.

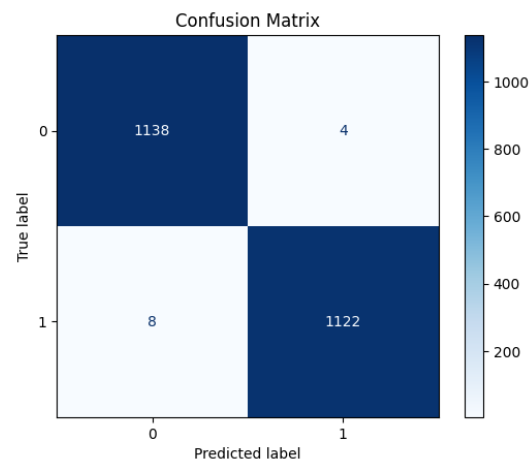
به منظور تحلیل عمیق تر خطاهای مدل، بررسی ماتریس درهم ریختگی نشان داد که در مجموع تنها دوازده نمونه از بیش از ۲۲۶۰ نمونه آزمون به اشتباه طبقه بندی شده اند و از میان آن ها، هیچ کدام مربوط به نمونه های مبهم شده نبوده و بیشتر

در شکل (۹)، نمودار هزینه مدل در طول فرایند آموزش ارائه شده است. مشابه نمودار صحت، بخش های سفید و آبی به ترتیب نشان دهنده غیرفعال بودن و فعال بودن وزن های مدل پایه برای آموزش هستند. کاهش مداوم هزینه مدل در طول دوره های آموزش، عملکرد مناسب و اثربخشی فرایند آموزش را تأیید می کند. جزئیات فرایند آموزش مدل در جدول (۵) ارائه شده است.

(جدول ۵): مشخصات فرایند آموزش مدل
(Table-5): Model Training Process Specifications

مجموع تعداد epochها		۳۰
تعداد epochها با غیرفعال بودن آموزش مدل پایه		۱۶
تعداد epochها با فعال بودن آموزش مدل پایه		۱۴
Train	میزان صحت در انتها	۰.۹۹۱
	میزان تابع هزینه در انتها	۰.۰۲
Validation	میزان صحت در انتها	۰.۹۹۳
	میزان تابع هزینه در انتها	۰.۰۱

نتایج به دست آمده نشان دهنده عملکرد مطلوب مدل پس از آموزش است. صحت مدل در داده های آموزشی به ۹۹.۱۳ درصد و در داده های اعتبارسنجی به ۹۹.۳۸ درصد رسیده که بیان کننده توانایی بالای مدل در یادگیری الگوها و تعمیم دهی به داده های دیده نشده و مقادیر پایین تابع هزینه نیز نشان دهنده خطای کم در پیش بینی ها است. این نتایج حاکی از یک مدل با کارایی بالا و تعمیم پذیری مناسب برای وظیفه طبقه بندی دودویی است؛ همچنین، برای ارزیابی دقیق تر، عملکرد مدل ها با استفاده از مجموعه داده آزمون که به طور مجزا بر اساس روش HoldOut تهیه و محاسبه شده است. نتیجه ارزیابی شامل صحت ۹۹.۳ درصد، دقت ۹۹.۸ درصد، فراخوانی ۹۹.۱ درصد و F-Measure برابر با ۹۹.۴ درصد را نشان می دهد. در شکل (۱۰) نیز ماتریس درهم ریختگی^۱ عملکرد مدل بر روی مجموعه داده آزمون ارائه شده است.



(شکل ۹): ماتریس درهم ریختگی مدل
(Figure-10): Model confusion matrix

بررسی ماتریس درهم ریختگی عملکرد بسیار مناسب مدل را بر روی داده های آزمون تأیید می کند. این داده ها که پیش تر

^۱ Confusion Matrix

مربوط به نمونه‌هایی با ساختارهای مرزی و یا رفتارهای مخلوط بودند. این تحلیل نشان می‌دهد که بیشتر خطاهای مدل مربوط به نمونه‌های مرزی است که در بسیاری از مدل‌های طبقه‌بندی نیز به صورت ذاتی رخ می‌دهد.

برای روشن‌تر شدن جایگاه مدل پیشنهادی نسبت به مطالعات پیشین، جدول (۶) مقایسه‌ای بین عملکرد پژوهش حاضر و مطالعات کلیدی مرتبط ارائه می‌دهد. دقت ۹۹.۳ درصد مدل پیشنهادی، از پژوهش‌های مشابه که با رویکردهای صوتی یا تصویری کار کرده‌اند (نظیر پژوهش [۲۵]) بالاتر است.

(جدول-۶): مقایسه عملکرد مدل پژوهش جاری با پژوهش‌های مرتبط

(Table-8): Comparing the performance of the current research model with related research

ردیف	پژوهش	نوع رویکرد		صحت	دقت	فراخوانی	امتیاز F	استفاده از مجموعه داده یکسان
		A*	I*					
۱	[۱۳]	*		۹۳	۹۲	۹۳	۹۲	بله
۲	[۲۰]	*		۹۵	۹۳	۸۹	۸۱	بله
۳	[۲۵]	*		۹۸.۹	۹۹.۰	۹۹.۶	۹۹.۳	بله
۴	پژوهش جاری	*	*	۹۹.۳	۹۹.۸	۹۹.۱	۹۹.۴	--

A*: رویکرد مبتنی بر داده‌های صوتی I*: رویکرد مبتنی بر داده‌های تصویری

از سوی دیگر، بدافزارهای امروزی با استفاده از روش‌های پیشرفتهٔ مبهم‌سازی، شناسایی را دشوارتر می‌کنند؛ هرچند مدل پیشنهادی توانست نمونه‌های موجود در مجموعه داده را با دقت کامل تشخیص دهد، اما احتمال ناکارآمدی مدل در برابر روش‌های جدیدتر وجود دارد و نیاز به بهبود مدل برای مقابله با این روش‌ها احساس می‌شود؛ همچنین، مجموعه دادهٔ تحلیل پویا با ابزار CopperDroid در شرایط خاصی جمع‌آوری شده و ممکن است استفاده از ابزارها و شرایط دیگر به جمع‌آوری داده‌های غنی‌تر و دقیق‌تر منجر شود؛ در نهایت، ارزیابی راه‌کار پیشنهادی با فرض صحت نتایج پژوهش‌های معتبر صورت گرفت و مقایسه‌ها با بهترین پژوهش‌هایی انجام شد که از مجموعه داده‌های مشترک استفاده کرده بودند، تا هم ارزش علمی مقایسه افزایش یابد و هم در زمان و هزینه صرفه‌جویی شود.

یکی دیگر از چالش‌های موجود در این پژوهش، خطر بیش‌برازش به دلیل آموزش و ارزیابی مدل تنها بر روی مجموعه دادهٔ CIC است؛ هرچند این مجموعه داده شامل بدافزارهای متنوع، نمونه‌های واقعی و مبهم‌شده است، اما تکیه بر تنها یک منبع داده می‌تواند دقت مدل را در شرایط واقعی و داده‌های ناشناخته کاهش دهد. برای کاهش این خطر، از تقسیم دقیق داده‌ها به آموزش، اعتبارسنجی و آزمون استفاده و همچنین آموزش مدل به صورت تدریجی^۱، با کمک فرایند یادگیری انتقالی انجام شد تا توان تعمیم مدل افزایش یابد؛ با این حال، برای اعتباربخشی بیشتر، در مراحل بعدی پژوهش، برنامه‌ریزی شده است تا مدل ارائه شده بر روی

نتایج این پژوهش نشان داد که راه‌کار پیشنهادی در تشخیص بدافزارهای اندرویدی عملکرد بسیار مناسبی دارد. نگاشت بابت‌های فایل‌های برنامه به حوزهٔ صوت و استخراج طیف‌های مختلف از آن، امکان شناسایی ویژگی‌های متنوع به وسیلهٔ مدل یادگیری عمیق را فراهم کرده و دقت تشخیص را به میزان قابل توجهی افزایش داده است.

با توجه به این نتایج، می‌توان گفت که رویکرد ارائه شده در این پژوهش نه تنها از نظر معماری مدل و ساختار داده نوآورانه است، بلکه در عمل نیز توانسته نسبت به روش‌های پیشین عملکرد بهتری ثبت کند و به دلیل ماهیت جعبه سیاه گونهٔ آن، برای شناسایی تهدیدات نوظهور، بسیار مناسب‌تر است.

۵- محدودیت و تهدیدهای وارد بر پژوهش

پژوهش حاضر، مانند هر پژوهش دیگری، با محدودیت‌ها و تهدیدهایی روبه‌رو بوده است؛ یکی از محدودیت‌های اصلی مربوط به فایل‌های APK ورودی بود؛ زیرا برخی از آن‌ها در مجموعه داده دچار خرابی بودند و ناچار باید حذف می‌شدند؛ به منظور حفظ اصالت داده‌ها و جلوگیری از استفاده از روش‌های نمونه‌سازی، تلاش شد با تعمیر فایل‌ها، بیشترین میزان ممکن از داده‌ها بازیابی و استفاده شود؛ البته افزایش حجم و تنوع مجموعه داده می‌تواند به بهبود عملکرد مدل کمک کند؛ همچنین محدودیت منابع پردازشی، استفاده از معماری‌های پیچیده‌تر شبکه‌های عصبی را زمان‌بر و پرهزینه می‌کند؛ بنابراین تعادلی منطقی میان زمان آموزش و عملکرد مدل ایجاد شد تا هم کارایی حفظ شود و هم مدل نهایی در کاربردهای واقعی، به منابع پردازشی کمتری نیاز داشته باشد.

^۱ Freezing & Unfreezing

مجموعه داده‌های دیگر از جمله MalGenome، Drebin و Androzoo نیز ارزیابی شود. این اقدام می‌تواند دید روشن‌تری از عملکرد مدل در مواجهه با تهدیدات متنوع‌تر فراهم کرده و پایداری آن را اثبات کند.

۶- نتیجه‌گیری و کارهای آینده

در این پژوهش، یک چهارچوب نوآورانه برای تشخیص بدافزارهای اندرویدی با تکیه بر تحلیل ایستا و یادگیری عمیق ارائه شد. رویکرد پیشنهادی با نگاشت بایت‌کد خام برنامه‌های اندرویدی به سیگنال‌های صوتی و استخراج سه طیف MFCC، BFCC و GFCC، داده‌ها را به صورت تصاویر رنگی سه‌کاناله بازنمایی کرد. این تصاویر به مدل از پیش آموزش‌دیده EfficientNetV2B0 به عنوان ورودی داده شدند و با بهره‌گیری از روش یادگیری انتقالی، فرایند آموزش مدل با دقت و کارایی بالا انجام شد. دقت نهایی ۹۹.۳ درصد، همراه با توانایی در تشخیص به‌طور کامل موفق برنامه‌های مبهم‌شده واقعی، نشان‌دهنده اثربخشی رویکرد پیشنهادی در شناسایی دقیق تهدیدهای اندرویدی است.

مزایای کلیدی روش پیشنهادی شامل مقاومت در برابر مبهم‌سازی، بی‌نیازی به مهندسی معکوس، کاهش پیچیدگی محاسباتی و قابلیت اجرا در محیط‌های عملیاتی واقعی است. استفاده از مدل از پیش آموزش‌دیده نیز زمان آموزش را کاهش داده و تعمیم‌پذیری مدل را بهبود بخشیده است. این عوامل روش ارائه‌شده را مناسب استفاده در سامانه‌های واقعی، ابزارهای قابل حمل و بازارهای اندرویدی می‌سازد.

در گام‌های آینده، افزایش تعداد داده‌ها و بهبود عملکرد مدل‌ها از اهداف اصلی است. با تمرکز بر بدافزارهای مبهم‌شده، تنوع داده‌ها افزایش خواهد یافت تا مدل در شناسایی این نوع بدافزارها نیز عملکرد بهتری داشته باشد؛ همچنین، ترکیب تحلیل ایستا و پویا در قالب یک مدل یادگیری عمیق واحد، فرایند آموزش را ساده‌تر و اطلاعات استخراج‌شده را غنی‌تر خواهد کرد؛ علاوه بر این، توسعه مدل‌های توضیح‌دهنده برای شناسایی بخش‌های مخرب فایل‌ها و ایجاد ابزارهای پیش‌بینی رفتار برنامه‌ها، از دیگر اهداف آینده است که می‌تواند امنیت کاربران را افزایش چشم‌گیری دهد.

در نهایت، تولید یک ابزار برون‌خط بر اساس مدل آموزش‌دیده، برای استفاده کاربران و بازارهای برنامه‌های اندرویدی در دستور کار است. با توجه به حجم مناسب مدل نهایی، امکان استفاده از آن بر روی گوشی‌های تلفن همراه فراهم است و می‌تواند ابزاری کارآمد و قابل اعتماد برای شناسایی بدافزارها باشد. این رویکردها نه تنها کارایی مدل را بهبود می‌بخشند، بلکه به کاربران اطمینان بیشتری در استفاده از برنامه‌های اندرویدی می‌دهند؛ با این حال، پژوهش حاضر نیز با برخی محدودیت‌ها مواجه بوده است؛ از جمله، ارزیابی مدل تنها بر

روی مجموعه داده CIC انجام و اگرچه شامل بدافزارهای واقعی و مبهم‌شده است، اما امکان دارد در مواجهه با انواع خاص دیگر بدافزارها، نیاز به تنظیم مجدد مدل وجود داشته باشد؛ همچنین، روش ارائه‌شده تنها مبتنی بر تحلیل ایستا بوده و رفتار زمان اجرای برنامه‌ها را به‌طور مستقیم در نظر نمی‌گیرد که این مسئله نیازمند انجام تحلیل پویا است.

در گام‌های آینده این مسیر پژوهشی، می‌توان موارد مختلفی را انجام داد؛ از جمله:

- ارزیابی روش پیشنهادی بر روی مجموعه داده‌های دیگر مانند Drebin، Androzoo و MalGenome برای بررسی پایداری مدل
- ترکیب تحلیل ایستا و پویا در یک چهارچوب یک‌پارچه: برای این هدف، از ابزارهایی مانند DroidBox یا TaintDroid برای استخراج رفتارهای زمان اجرا استفاده خواهد شد و این داده‌ها می‌توانند برای آموزش مدل‌های یادگیری عمیق استفاده شوند.
- به‌کارگیری مدل‌های قابل توضیح^۱ برای شناسایی نواحی مؤثر در تصمیم‌گیری مدل و ارائه شفافیت بیشتر در دلایل تشخیص
- توسعه یک ابزار برون‌خط و سبک برای کاربران عادی و توسعه‌دهندگان، با قابلیت اجرای مدل بر روی گوشی‌های هوشمند
- در مجموع، پژوهش حاضر با بهره‌گیری از تلفیق هوشمندانه تحلیل ایستا، پردازش صوتی، تصویرسازی رنگی از داده‌های طیفی و یادگیری انتقالی، روشی دقیق، مقاوم و عملیاتی برای تشخیص بدافزارهای اندرویدی ارائه داده است که می‌تواند گامی مؤثر در افزایش امنیت کاربران و ارتقای سطح دفاع سایبری در سکوی اندروید به‌شمار آید.

6-References

۶-مراجع

- [1] J. Li et al., "Significant permission identification for machine-learning-based Android malware detection," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3216–3225, 2018.
- [2] N. Peiravian and X. Zhu, "Machine learning for android malware detection using permission and api calls," in *Proc. IEEE 25th Int. Conf. Tools with Artificial Intelligence (ICTAI)*, Washington, DC, USA, 2013, pp. 556–560.
- [3] Y. Zhou et al., "Taming information-stealing smartphone applications (on Android)," in *Trust and Trustworthy Computing: 4th Int. Conf.*, TRUST 2011, Pittsburgh, PA, USA, June 22–24, 2011, vol. 6700, pp. 423–434.
- [4] H.-J. Zhu et al., "DroidDet: effective and robust detection of Android malware using static analysis along with rotation forest model," *Neurocomputing*, vol. 272, pp. 638–646, 2018.
- [5] V. M. Afonso et al., "Identifying Android malware using dynamically obtained features," *J. Comput. Virol. Hacking Tech.*, vol. 11, pp. 9–17, 2015.

¹ Explainable AI

- [24] A. Azab and M. Khasawneh, "Msic: malware spectrogram image classification," *IEEE Access*, vol. 8, pp. 102007–102021, 2020.
- [25] P. Tarwireyi, A. Terzoli, and M.O. Adigun, "Using multi-audio feature fusion for android malware detection," *Computers & Security*, vol. 131, p. 103282, 2023.
- [26] T. Irino and R.D. Patterson, "A time-domain, level-dependent auditory filter: The gammachirp," *The Journal of the Acoustical Society of America*, vol. 101, no. 1, pp. 412–419, 1997.
- [27] P. Tarwireyi, A. Terzoli, and M.O. Adigun, "BarkDroid: Android malware detection using bark frequency Cepstral coefficients," *Indonesian Journal of Information Systems*, vol. 5, no. 1, pp. 48–63, 2022.
- [28] X. Zhao and D. Wang, "Analyzing noise robustness of MFCC and GFCC features in speaker identification," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, BC, Canada, 2013.
- [29] A. G. Howard et al., "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.
- [30] K. Simonyan, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
- [31] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [32] M. Tan and Q. V. Le, "Efficientnetv2: Smaller models and faster training," in *Proc. Int. Conf. Machine Learning (ICML)*, PMLR, 2021.
- [33] M. Ghasemi, A. Horri, and M. E. Basiri, "Detecting Android malware with offloading approach in cloud computing," *J. Signal & Data Processing.*, vol. 21, no. 3, pp. 137-148, 2024.
- [34] M. Deypir, "RiskMeter: A Tool for Measuring Precise Security Risk Values of Mobile Device Applications," *J. Signal & Data Process.*, vol. 14, no. 3, pp. 23–36, 2017.
- [35] P. Yan and Z. Yan, "A survey on dynamic mobile malware detection," *Softw. Qual. J.*, vol. 26, no. 3, pp. 891–919, 2018.
- [7] G. Canfora, F. Mercaldo, and C. A. Visaggio, "Mobile malware detection using op-code frequency histograms," in *2015 12th Int. Joint Conf. on e-Business and Telecommunications (ICETE)*, 2015, pp. 1–7.
- [8] M. K. Alzaylaee, S. Y. Yerima, and S. Sezer, "DL-Droid: Deep learning based Android malware detection using real devices," *Computers & Security*, vol. 89, p. 101663, 2020.
- [9] J. Kim et al., "MAPAS: a practical deep learning-based Android malware detection system," *Int. J. Inf. Secur.*, vol. 21, no. 4, pp. 725–738, 2022.
- [10] X. Xiao et al., "Android malware detection based on system call sequences and LSTM," *Multimed. Tools Appl.*, vol. 78, pp. 3979–3999, 2019.
- [11] S. I. Imtiaz et al., "DeepAMD: Detection and identification of Android malware using high-efficient Deep Artificial Neural Network," *Future Gener. Comput. Syst.*, vol. 115, pp. 844–856, 2021.
- [12] D. T. Dehkordy and A. Rasoolzadegan, "A new machine learning-based method for Android malware detection on imbalanced dataset," *Multimed. Tools Appl.*, vol. 80, pp. 24533–24554, 2021.
- [13] L. Shen et al., "Self-attention based convolutional-LSTM for Android malware detection using network traffics grayscale image," *Appl. Intell.*, 2022, pp. 1–23.
- [14] K. Bakour and H. M. Ünver, "DeepVisDroid: Android malware detection by hybridizing image-based features with deep learning techniques," *Neural Comput. Appl.*, vol. 33, pp. 11499–11516, 2021.
- [15] Y. Ding et al., "Android malware detection method based on bytecode image," *J. Ambient. Intell. Humaniz. Comput.*, 2020, pp. 1–10.
- [16] T. Hsien-De Huang and H.-Y. Kao, "R2-d2: Color-inspired convolutional neural network (CNN)-based Android malware detections," in *Proc. IEEE Int. Conf. on Big Data (Big Data)*, Seattle, WA, USA, 2018, pp. 2017–2026.
- [17] D. Vasan et al., "IMCFN: Image-based malware classification using fine-tuned convolutional neural network architecture," *Comput. Networks*, vol. 171, p. 107138, 2020.
- [18] F. Mercaldo and A. Santone, "Deep learning for image-based mobile malware detection," *J. Comput. Virol. Hacking Tech.*, vol. 16, no. 2, pp. 157–171, 2020.
- [19] P. Yadav et al., "EfficientNet convolutional neural networks-based Android malware detection," *Computers & Security*, vol. 115, p. 102622, 2022.
- [20] J. Feng et al., "A two-layer deep learning method for Android malware detection using network traffic," *IEEE Access*, vol. 8, pp. 125786–125796, 2020.
- [21] M. Farrokhmanesh and A. Hamzeh, "Music classification as a new approach for malware detection," *J. Comput. Virol. Hacking Tech.*, vol. 15, pp. 77–96, 2019.
- [22] F. Mercaldo, A. Santone, "Audio signal processing for android malware detection and family identification," *J. Comput. Virol. Hacking Tech.*, vol. 17, no. 2, pp. 139–152, 2021.
- [23] R. Casolare et al., "Mobile Family Detection through Audio Signals Classification," in *Proc. SECUREPT*, 2021.

علی علیایی **طرقه مدرک کارشناسی**
و کارشناسی ارشد خود را در رشته
مهندسی کامپیوتر گرایش نرم افزار
به ترتیب در سال های ۱۴۰۰ و ۱۴۰۳ از
دانشگاه فردوسی مشهد دریافت کرد.
ایشان در حال حاضر دانشجوی دکترای مهندسی کامپیوتر
گرایش نرم افزار در دانشگاه فردوسی مشهد است. زمینه های
پژوهشی مورد علاقه ایشان مهندسی نرم افزار، هوش
مصنوعی و یادگیری عمیق است.
نشانی رایانامه ایشان عبارت است از:
ali.olyaei@mail.um.ac.ir





عباس رسولزادگان کارشناسی خود را

در رشته مهندسی کامپیوتر - مهندسی

نرم افزار از دانشگاه هوانوردی تهران در

سال ۱۳۸۳، کارشناسی ارشد را در رشته

مهندسی کامپیوتر - مهندسی نرم افزار از

دانشگاه صنعتی امیرکبیر در سال ۱۳۸۶ و دکترای خود را

در رشته مهندسی کامپیوتر - نرم افزار از همان دانشگاه در

سال ۱۳۹۲ دریافت کرد. ایشان در حال حاضر دانشیار گروه

مهندسی کامپیوتر دانشگاه فردوسی مشهد است. زمینه های

پژوهشی مورد علاقه ایشان عبارت اند از: مهندسی کیفیت

نرم افزار و تست هوشمند نرم افزار.

نشانی رایانامه ایشان عبارت است از:

rasoolzadegan@um.ac.ir

برای اطلاعات بیشتر درباره ایشان، به تارنمای SQLab

مراجعه کنید:

<https://SQLab.um.ac.ir>

