تشخیص بدافزارهای اندرویدی با استفاده از روش یادگیری ترکیبی پشته‌ای

مونا زارع[۱]، علیرضا رضوانیان[۱]

[۱]گروه مهندسی کامپیوتر، دانشگاه علم و فرهنگ، تهران
monazare754@gmail.com، rezvanian@usc.ac.ir

چکیده

بدافزارها، برنامه‌هایی هستند که رفتار مخربی دارند و برای آسیب رساندن به سیستم‌ها و شبکه‌های کامپیوتری طراحی شده‌اند. بدافزارها، تنوعی از کارهای مخرب از سرقت اطلاعات حساس تا از بین بردن کل سیستم‌ها را با اهداف مختلفی چون تجاری، اجتماعی، اقتصادی، سیاسی، نظامی یا شخصی را انجام می‌دهند. با توجه به پیچیده‌تر شدن ساختار و رفتار بدافزارها در سیستم‌های اندروید، رویکردهای سنتی، تکرارگرا و آماری عملاً کارساز نبوده و در گذر زمان منسوخ شده است. در این مقاله، بعد از آماده سازی و نرمالسازی داده‌ای، از یک روش یادگیری ترکیبی به صورت پشته‌سازی برای تشخیص بدافزارهای اندرویدی استفاده شده است؛ به این صورت که درخت تصمیم (DT)، بیز ساده (NB) و رگرسیون خطی (LR) به‌عنوان یادگیرنده‌های ضعیف و ماشین بردار پشتیبان (SVM) به‌عنوان یادگیرنده قوی منظور شده است. نتایج آزمایش‌ها بر روی داده‌های سیستم‌های اندرویدی براساس معیارهای دقت، بازخوانی و صحت حاکی از بهبود نتایج روش پیشنهادی نسبت به نتایج روش‌های پایه و چند روش اخیر دارد.

کلمات کلیدی: تشخیص بدافزار، اندروید، یاد گیری ماشین، پشته‌سازی، درخت تصمیم، ماشین بردار پشتیبان.

۱ مقدمه

فایل‌های اجرایی مخرب، برنامه‌هایی هستند که جهت نفوذ و آسیب به شبکه‌های کامپیوتری بدون اجازه و اطلاع کاربر طراحی شده‌اند و امروزه به‌عنوان یک تهدید جدی برای امنیت این شبکه‌ها به شمار می‌روند. این برنامه‌های مخرب که در اصطلاح بدافزار نامیده می‌شوند، قادر هستند که انواع کارهای مخرب از سرقت اطلاعات حساس گرفته تا از بین بردن کل سیستم‌ها یا دستگاه‌ها را انجام دهند. نفوذ بدافزارها در اکثر حملات سایبری از طریق سرقت داده‌ها، منجر به سرقت هویت و حتی نقض گسترده داده‌ها می‌گردد. امروزه بدافزارها با استفاده از روش‌های مبهم‌سازی، پیچیده‌تر شده به گونه‌ای که تشخیص آن‌ها نیز دشوارتر شده

است. بنابراین، بدافزارها و تهدیدهای آن‌ها یکی از بزرگترین چالش‌ها در امنیت شبکه‌های کامپیوتری به شمار می‌رود [۱]. یکی از مشکلات اساسی برای درک صحیح رفتارهای مخرب و نسخه‌های جدید در توسعه بدافزارها، تغییرات زیاد آن‌ها است. از این‌رو، روش‌های سنتی مانند تطابق چند رشته کد از امضای بدافزارها به تنهایی کافی نیستند. استفاده از روش‌های مبتنی بر امضا نیز کند و گران هستند و در مقابل فایل‌های مخرب ناشناخته و دستکاری شده مؤثر نیستند [۲]. بنابراین، همزمان با توسعه‌ی بدافزارها و پیچیده‌تر شدن فرآیند تشخیص آن‌ها، روش‌های سنتی مبتنی بر آمار و مبتنی بر امضا نیز نمی‌تواند به خوبی عمل کند، که تلاش برای استفاده از روش‌های مبتنی بر یادگیری ماشین وجود دارد.

در این مقاله یک روش یادگیری ترکیبی پشته ساز با ترکیب درخت تصمیم، بیز ساده و رگرسیون خطی به عنوان یادگیرنده‌های ضعیف و ماشین بردار پشتیبان به عنوان یادگیرنده قوی به منظور تشخیص بدافزارهای اندرویدی طراحی شده است. در بخش پایانی، ارزیابی روش پیشنهادی در مقایسه با چند روش پایه و اخیر گزارش خواهد شد.

۲ پیشینه‌ی پژوهش

به صورت کلی دو روش، به صورت روش‌های ایستا و روش‌های پویا برای استخراج رفتار بدافزار وجود دارد. در روش‌های ایستا، کد باینری بدافزار بدون اینکه اجرا شود، (مثلا براساس گراف کنترل جریان و گراف فراخوانی توابع) استفاده شده است. بزرگ‌ترین مزیت روش‌های ایستا این است که به دلیل طی کردن تمامی حالت‌های ممکن، مشخص می‌شود که بدافزار مورد نظر در شرایط غیر معمول به چه شکلی رفتار خواهد کرد. همچنین چون در این روش بدافزار در هیچ سیستمی اجرا نمی‌شود، خطر آلودگی سیستم میزبان حداقل ممکن است. بزرگترین عیب روش ایستا این است که ممکن است در مقابل مبهم‌سازی دچار مشکل شود و در تشخیص درست عمل نکند. در روش‌های پویا، تشخیص بدافزار در زمان اجرای آن صورت می‌گیرد و در برابر مبهم‌سازی کد مقاوم است، بزرگترین مشکل روش‌های پویا این است که چون در یک محیط و یک بار اجرا می‌شود، فقط همان یک مسیر بررسی می‌شود و مسیرهای دیگر قابل تشخیص نخواهد بود. برای رفع این مشکل روش، بدافزار را در محیط‌های مختلف و روی داده‌های مختلف چندین بار اجرا می‌کنند [۳]. در ادامه، برخی از روشهای معروف ارائه شده در سال‌های اخیر معرفی شده است.

در مرجع [۴]، با هدف پیش‌گیری از ورود به حریم خصوصی و سرقت اطلاعات حساس در دستگاه‌های تلفن همراه، یک چارچوب یادگیری تجمعی انباشته به‌نام SEDMDroid برای تشخیص بدافزارها ارائه شده است. هم‌چنین، روش ماشین بردار پشتیبان به منظور طبقه‌بندی تلفیقی به کار برده شده است تا اطلاعات تکمیلی ضمنی را از خروجی اعضای گروه بیاموزد و نتیجه پیش‌بینی نهایی را ارائه دهد. نتایج شبیه‌سازی بیان‌گر دقت بیش از ۹۰ درصدی روش SEDMDroid در تشخیص بدافزار نسبت به مجموعه ارزیابی است. در [۵]، برای طبقه‌بندی برنامه‌های مخرب اندروید، استفاده از ترکیب شبکه‌های عصبی بازگشتی و پیچشی پیشنهاد شده است که هدف آن، یادگیری ارتباط کلی بین الگوهای رشته مبهم از نام بسته برنامه و نام صاحب گواهی است. این مدل ویژگی‌های یادگیری ماشین را استخراج می‌کند و یک واحد شبکه عصبی

پیچشی اضافی نیز فرایند استخراج ویژگی را بهبود می‌دهد. نتایج شبیه سازی بیانگر این است که رویکرد ترکیبی نسبت به مدل‌های مبتنی بر Ngram از کارایی بالاتری برخوردار است.

در مرجع [۶]، نویسندگان دو روش مبتنی بر یادگیری تجمعی با داده‌های مقیاس بزرگ طراحی و ارائه کرده‌اند. روش اول براساس استراتژی رای‌گیری وزنی یادگیری تجمعی استوار است و روش دوم مجموعه بهینه‌ای از طبقه‌بندی کننده‌های اصلی را برای انباشت انتخاب می‌کند. نتایج شبیه‌سازی اثربخشی روش‌های ترکیبی را تایید می‌کند. در مرجع [۷]، رویکردی با استفاده از هم‌افزایی ویژگی‌های شمارنده‌های سخت‌افزاری و طبقه‌بندی شبکه عصبی پرسپترون چندلایه بهینه ارائه شده است. در این روش استخراج ویژگی‌هایی با قابلیت تفکیک‌پذیری بالا و نیز از شبکه عصبی بهینه شده بوسیله‌ی الگوریتم سنجاقک، استفاده داده شده است. نتایج حاصل از شبیه‌سازی‌ها، کارایی بالاتر طبقه‌بندی ارائه شده برای تشخیص فایل‌های آلوده شده به بدافزار را نشان می‌دهد. در مرجع [۸]، روشی جهت تشخیص بدافزارها با استفاده از رویکرد داده کاوی معرفی شده است. ایده اصلی روش پیشنهادی ترکیب انتخاب رای اکثریت برای دسته‌بندی و میانگین مقادیر تخمینی است. که در این راستا، روش ارائه شده بیانگر دقت بالای دسته‌بندی و نرخ صحیح تشخیص بدافزارها به بالای ۹۸ درصد است. در مرجع [۹]، روشی ارائه شده که هدف آن، تشخیص عوامل و ویژگی‌ها به صورت ایستا است و همچنین به کمک یک سیستم تصمیم‌یار هوشمند به تشخیص و هشدار این بدافزارها، پرداخته شده است. دقت روش ارائه شده در تشخیص بدافزارها بیش از ۹۷ درصد است. در مرجع [۱۰]، تمرکز بر تشخیص بدافزار از طریق مقایسه‌ی اطلاعات موجود در ساختارهای داده فضای حافظه کاربر است. برای تسریع و تضمین صحت اطلاعات استخراج شده، هم‌زمان از اطلاعات موجود در چندین ساختار مدیریت حافظه در فضای کاربر و هسته استفاده می‌گردد. سپس، برای ارزیابی ویژگی‌های استخراج شده، نمونه‌ها براساس ویژگی‌های انتخاب شده دسته‌بندی می‌شوند. بهترین نتایج شامل نرخ تشخیص ۹۸% و نرخ مثبت کاذب ۱۷% هستند.

۳ روش پیشنهادی

در این مقاله یک روش یادگیری ترکیبی به صورت پشته‌ای شامل ۵ مرحله به شرح زیر ارائه می‌شود.

۱.۳ پیش‌پردازش بر روی داده‌ها

در این مرحله، نمونه‌های پرت از داده‌ها حذف می‌گردد. برای پاک‌سازی داده‌ها، حذف نمونه‌های پرت و داده‌های غیر مرتبط مدنظر است. داده‌های غیرمرتبط، سطرهایی از مجموعه داده است که تهی بوده یا دارای مقدار نامشخص است. همچنین، در این مقاله، از روش پاک‌سازی داده‌ها استفاده شده است، بدین صورت که داده‌ها مورد بررسی قرار گرفته تا در صورتی‌که سطر یا ستونی دارای مقادیر تهی یا غیرمرتبط است، مشخص گردد. سپس مقادیر قبل و بعد از نمونه‌ای که دارای مقدار تهی یا غیرمرتبط است را مورد بررسی قرار داده و میانگین آن‌ها جایگزین می‌گردد.

۲.۳ آماده‌سازی داده‌ها

پس از حل مشکل نمونه‌های پرت، آماده‌سازی داده‌ها انجام گیرد. بدین منظور، داده‌های پیش‌پردازش شده به قالب قابل قبول برای استفاده در ابزارهای موردنیاز تبدیل می‌شود تا در سرورهای اصلی شبکه و در پس‌زمینه، کل داده‌ها که به منظور آموزش روش پیشنهادی جهت تشخیص بدافزارهای تروجانی استفاده شده، به یک فرمت قابل قبول برای نرم‌افزارها و ابزارهای شبکه تبدیل شوند. پس از اینکه به‌صورت سطحی مجموعه داده، مورد بررسی قرار داده شد، در صورت بایستی نرمال‌سازی انجام شود.

۳.۳ نرمال‌سازی داده‌ها

در این مرحله، مقادیر هر ویژگی استفاده شده از مجموعه داده بین ۰ تا ۱ نرمال شده، سپس کلیه مجموعه داده در قالب یک ماتریس نگاشت شده و با تغییر سطرهای ماتریس، عملیات نرمال‌سازی صورت می‌گیرد. برای نرمال‌سازی مقادیر هر مجموعه داده پیوسته، از رابطه (۱) استفاده شده است.

$$Normalize(x) = \frac{x - X_{minP+}}{X_{max} - X_{min}} \quad (۱)$$

به طوریکه X_{max} و X_{min} به ترتیب مقدار بیشینه و کمینه در دامنه‌ی ویژگی X است. پس از نرمال‌سازی داده‌ها، مقادیر کلیه ویژگی‌ها در بازه‌ی $[۰, ۱]$ قرار گرفته و یک پیش‌پردازش به روی نمونه‌های موجود اعمال شده و داده‌هایی که دارای مقادیر پرت استفاده هستند، حذف می‌گردند. علاوه‌بر این، نمونه‌هایی که هیچ فعالیت و عملکردی از آن‌ها ثبت نشده است، از مجموعه داده اصلی پاک‌سازی می‌شود. برای نرمال‌سازی مجموعه داده گسسته از رابطه (۲) استفاده می‌شود.

$$Z_{iF} = \frac{r_{iF-۱}}{M_{F-۱}} \quad (۲)$$

۴.۳ حذف نمونه‌های پرت

در این مرحله با کمک الگوریتم خوشه‌بندی DBScan، داده‌ها تفکیک شده و داده های پرت حذف می‌شود تا بتوان از نتایج مناسبی برخوردار شد.

۵.۳ سیستم یادگیری ترکیبی پشته‌ساز

در روش یادگیری پشته‌ساز پیشنهادی از سه الگوریتم یادگیری ماشین درخت تصمیم (DT)، بیز ساده (NB) و لاجستیک رگرسیون (LR) به عنوان یادگیرنده ضعیف استفاده می‌گردد و خروجی این سه الگوریتم به همراه داده اولیه به‌عنوان ورودی الگوریتم ماشین بردار پشتیبان (SVM) به عنوان یادگیرنده قوی بکار گرفته می‌شود. ساختار کلی این سیستم پشته‌سازی در شکل ۱ نمایش داده شده است.

همانطور که در شکل ۱ مشاهده می‌گردد، مجموعه داده ابتدا به الگوریتم‌های یادگیری ماشین درخت تصمیم، الگوریتم بیز ساده و الگوریتم لاجستیک رگرسیون وارد می‌شود. داده‌های آموزشی به‌عنوان ورودی این الگوریتم‌ها جهت آموزش و تولید مدل استفاده می‌شود. سپس خروجی مدل این سه الگوریتم و داده‌های آزمایشی به عنوان داده‌های آموزشی به ماشین بردار پشتیبان برای تولید مدل نهایی جهت تشخیص بدافزار، داده می‌شود. بطور کلی مراحل انجام روش پشته‌ساز پیشنهادی به شرح ذیل است:

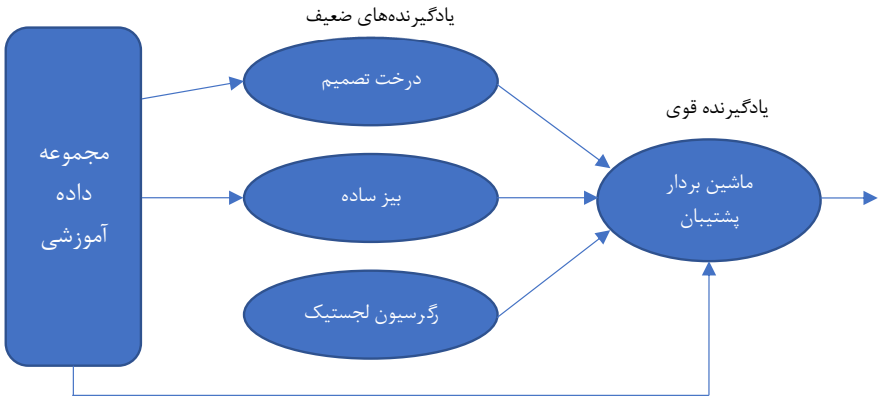
۱. ابتدا مجموعه داده مربوط به بدافزارها به سیستم پشته‌ساز به‌عنوان ورودی داده می‌شود. لازم به ذکر است که مجموعه داده استفاده شده از قبل به دو بخش آموزشی و آزمایشی تقسیم‌بندی شده است.

۲. نمونه‌های آموزشی به هسته اصلی الگوریتم درخت تصمیم با رویکرد آنتروپی، بیز ساده با رویکرد تحلیل گسسته و رگرسیون خطی داده می‌شود. در این مرحله هر کدام از الگوریتم‌های ذکر شده به ترتیب مدل خود را بر اساس نمونه‌های آموزشی دریافتی آموزش داده و مدلی را تولید می‌کنند. بنابراین، تا این مرحله سه مدل مربوط به درخت تصمیم، بیز ساده و رگرسیون خطی تولید شده است.

۳. در مرحله بعد خروجی سه مدل تولید شده به همراه داده اولیه به‌عنوان ورودی به الگوریتم ماشین بردار پشتیبان دو کلاسه با هسته داده شده و ماشین بردار پشتیبان نیز مدل برداری خود را بر اساس این سه مدل تولید می‌کند.

۴. پس از تولید کلیه مدل‌ها، نمونه‌های آزمایشی به هر کدام از مدل‌ها وارد شده و جداگانه طبقه‌بندی می‌شوند.

۵. در نهایت مدل نهایی پشته‌ساز تصمیم می‌گیرد که نمونه مورد نظر که از نمونه‌های آزمایشی وارد شده است یک نمونه نرمال یا بدافزار است.



شکل ۱: روش پشته‌ساز پیشنهادی جهت تشخیص بدافزار

۴ ارزیابی

در این بخش، ابتدا مجموعه داده‌ها، سپس معیارهای ارزیابی و در نهایت نتایج شبیه‌سازی آزمایش‌ها ارائه می‌شود.

۱.۴ مجموعه داده

در این مقاله از مجموعه داده بدافزارهایی که امنیت سیستم‌های اندروید را تحت تاثیر قرار می‌دهد، استفاده شده است. تعداد نمونه‌های مورد استفاده برابر با ۲۳۲.۱۷ نمونه و تعداد ویژگی‌ها برابر با ۸۶۹ ویژگی است[۱]. مجموعه داده به دو دسته داده‌های آموزشی و آزمایشی تقسیم شده است. ۶۰% از کل داده‌ها به‌عنوان داده‌های آموزشی و ۴۰% دیگر به‌عنوان داده‌های آزمایشی در نظر گرفته شده است.

۲.۴ معیارهای ارزیابی

معیارهای مورد استفاده جهت ارزیابی روش پیشنهادی به صورت Precision، Recall، Accuracy عبارتند از:

$$Precision = \frac{TP}{(TP + FP)} \quad (۳)$$

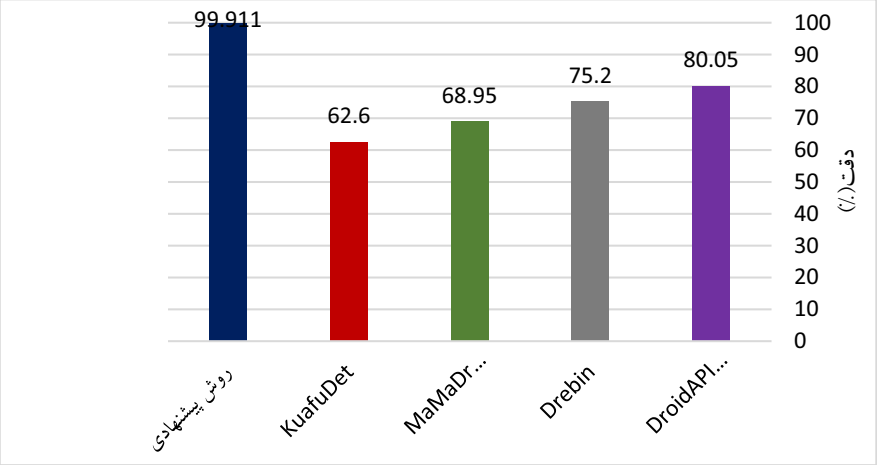
$$Recall = \frac{TP}{(TP + FN)} \quad (۴)$$

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \quad (۵)$$

به طور یکه پارامتر TP بیانگر تعداد نمونه‌هایی است که به درستی، بدافزار، تشخیص داده شده‌اند. پارامتر FP نیز بیانگر تعداد نمونه‌هایی است که به اشتباه، بدافزار، تشخیص داده شده‌اند. پارامتر FN بیانگر تعداد نمونه‌هایی است که به اشتباه نرمال تشخیص داده شده‌اند. TN بیانگر تعداد نمونه‌هایی است که به‌درستی نرمال تشخیص داده شده‌اند.

۳.۴ مقایسه نتایج روش پشته‌ساز پیشنهادی با سایر روش‌ها

برای مقایسه نتایج روش پیشنهادی، پیاده‌سازی انجام شده در محیط نرم‌افزاری MATLAB نسخه R2022b در شرایط یکسان با نتایج [۴]، [۱۱] استفاده شده است. در شکل ۲ دقت تشخیص بدافزار با استفاده از روش



شکل ۲: مقایسه دقت تشخیص بدافزار روش پشته‌ساز پیشنهادی با سایر روش‌ها

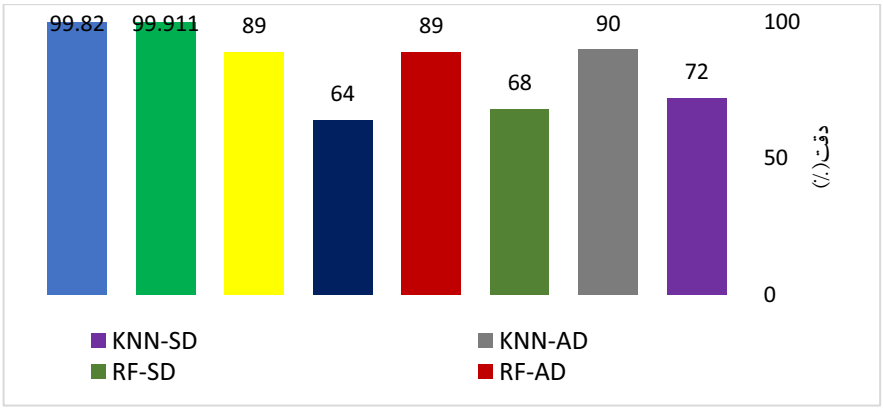
پشته‌ساز پیشنهادی و سایر روش‌هایی همچون KuafuDet، MaMaDroid، Drebin و DroidAPIMiner که مبتنی بر طبقه‌بندی و یادگیری ماشین نیستند، نشان داده شده است.

با توجه به نتایج شکل ۲، روش پیشنهادی توانسته نسبت به روش‌های KuafuDet، MaMaDroid، Drebin و DroidAPIMiner به ترتیب ۳۷٫۳۱%، ۳۰٫۹۶%، ۲۴٫۷۱% و ۱۹٫۸۶% دقت را بهبود ببخشد، که دقت بهبود قابل توجهی داشته است. در شکل ۴ دقت تشخیص بدافزار با استفاده از روش پشته‌ساز پیشنهادی و سایر روش‌های دیگر همچون KNN، Random forest و SVM، نشان داده شده است. در شکل ۳ روش‌های نام برده شده، به دو دسته کلی AD و SD تقسیم‌بندی شده‌اند. SD بیانگر روش‌هایی است که به صورت تشخیص خصمانه عمل می‌کنند و روش‌های AD مبتنی بر تشخیص بدون استراتژی خصمانه است که در مقاله [۱۱] مورد بررسی قرار گرفته است. همانطور که در شکل ۳ مشاهده می‌گردد، میانگین دقت روش پیشنهادی با سطح اجرا و بدون سطح اجرا برابر با ۹۹٫۸۶% است. میزان بهبود دقت تشخیص بدافزار در روش پیشنهادی نسبت به روش‌های SVM-AD، SVM-SD، RF-AD، RF-SD، KNN-AD و KNN-SD به ترتیب برابر با ۱۰٫۸۶%، ۳۵٫۸۶%، ۱۰٫۸۶%، ۳۱٫۸۶%، ۹٫۸۶% و ۲۷٫۸۶% است. از نتایج بدست آمده می‌توان این‌گونه استنتاج کرد که روش‌هایی که بدون تشخیص خصمانه عمل می‌کنند دارای دقت کمتری نسبت به روش‌هایی هستند که با تشخیص خصمانه عمل می‌کنند.

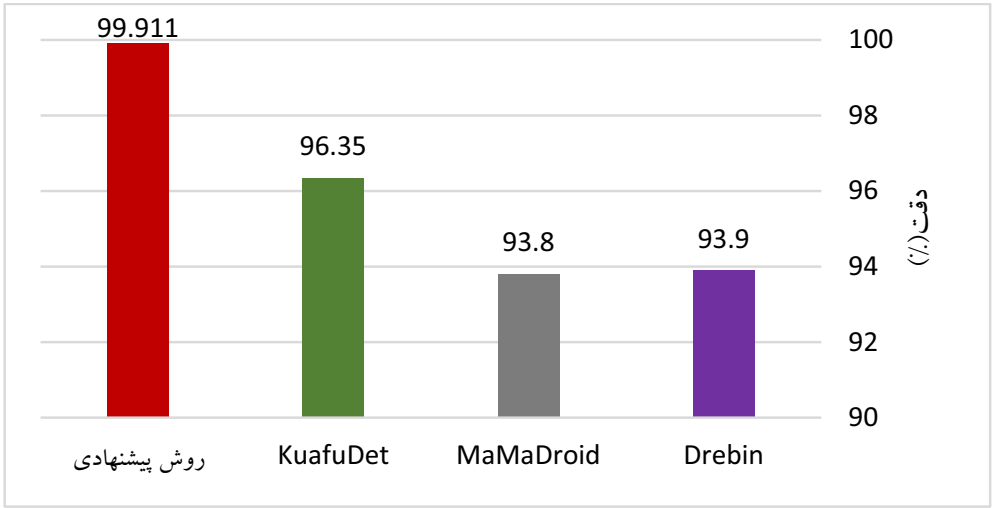
در شکل ۴ دقت روش پیشنهادی و روش‌های KuafuDet، MaMaDroid و Drebin با استفاده از الگوریتم‌های انتخاب ویژگی به منظور تشخیص بدافزارها نشان داده شده است.

همانطور که از نتایج شکل ۴ مشاهده می‌گردد، میزان دقت روش پیشنهادی با سطح اجرا و الگوریتم شبکه عصبی عمیق GMDH برابر با ۹۹٫۹۱% است. از این رو، میزان دقت روش پیشنهادی نسبت به روشهای KuafuDet، MaMaDroid و Drebin به منظور تشخیص بدافزارها به ترتیب برابر با ۳٫۵۶%،

[1]https://ieee-dataport.org/documents/dataset-malwarebeningn-permissions-android



شکل ۳: مقایسه دقت تشخیص بدافزار روش پشته‌ساز پیشنهادی با سایر روش‌های دیگر



شکل ۴: مقایسه دقت تشخیص بدافزار روش پیشنهادی با روش‌های KuafuDet، MaMaDroid و Drebin

جدول ۱: مقایسه دقت روش پیشنهادی با سایر روش‌های دیگر و روش SEDMDroid

روش	ApkAuditor	ACTS	DroidOL	NSCG	OmniDroid	SEDMDroid
دقت (%)	۸۸	۸۷٫۰۹	۸۴٫۲۶	۸۷٫۰۳	۸۹٫۰۷	۹۱٫۹۹

۶٫۱۱% و ۶٫۰۱% است. در جدول ۱، دقت روش پیشنهادی با روش SEDMDroid مطرح شده در مرجع [۴] که برای تشخیص بدافزار در دستگاه‌های اندروید استفاده شده و سایر روش‌های دیگر مقایسه شده است، که دقت روش پیشنهادی جهت تشخیص بدافزارها برابر با ۹۹% ۹۱. است. روش پیشنهادی در مقایسه با روش‌های ،ApkAuditor، ACTS، DroidOL، NSCG OmniDroid و SEDMDroid توانسته دقت را به ترتیب ۱۱٫۹۱%، ۱۲٫۰۱%، ۱۵٫۶۵%، ۱۲٫۶۱%، ۱۰٫۲۱% و ۷٫۹۱% بهبود دهد.

۵ نتیجه‌گیری

بدافزار به هر نرم‌افزار رایانه‌ای گفته می‌شود که رفتار مخرب داشته باشد و به رایانه میزبان صدمه بزند و با توجه به پیچیدگی رفتار بدافزارها، تشخیص بدافزارهای اخیر با روش‌های سنتی، مبتنی بر فراوانی و مبتنی بر الگو به سادگی امکان پذیر نیست. بنابراین در این مقاله، یک روش ترکیبی پشته سازی با ترکیب درخت تصمیم، رگرسیون خطی و بیز ساده به عنوان یادگیرنده های ضعیف و ماشین بردار پشتیبان به عنوان یادگیرنده قوی ارائه شد. نتایج شبیه سازی بر روی داده‌های سیستم‌های اندرویدی در مقایسه با چند روش پایه و اخیر حاکی از بهبود نسبی نتایج روش پیشنهادی است. به عنوان کارهای آینده شاید بتوان از روش‌های نوین یادگیری عمیق بهره برد.

مراجع

[1] M. S. Rana and A. H. Sung, “Evaluation of Advanced Ensemble Learning Techniques for Android Malware Detection,” Vietnam Journal of Computer Science, vol. 7, no. 2, pp. 145–159, 2020.

[2] A. Mahindru and A. L. Sangal, “MLDroid—framework for Android malware detection using machine learning techniques,” Neural Computing and Applications, vol. 33, no. 10, pp. 5183–5240, 2021.

[3] A. Damodaran, F. D. Troia, C. A. Visaggio, T. H. Austin, M. Stamp, “A comparison of static, dynamic, and hybrid analysis for malware detection.” Journal of Computer Virology and Hacking Techniques, vol. 13, pp. 1-12, 2017.

[4] H. Zhu, Y. Li, R. Li, J. Li, Z. You, and H. Song, “SEDMDroid: An Enhanced Stacking Ensemble Framework for Android Malware Detection,” IEEE Trans. Netw. Sci. Eng., vol. 8, no. 2, pp. 984–994, 2021.

[5] W. Y. Lee, J. Saxe, and R. Harang, “SeqDroid: Obfuscated android malware detection using stacked convolutional and recurrent neural networks,” Adv. Sci. Technol. Secur. Appl., pp. 197–210, 2019.

[6] D. Gupta and R. Rani, “Improving malware detection using big data and ensemble learning,” Comput. Electr. Eng., vol. 86, p. 106729, 2020.

[7] F. Idrees, M. Rajarajan, M. Conti, T.M. Chen and Y. Rahulamathavan, “PIndroid: A novel Android malware detection system using ensemble learning methods,” Elsevier, vol. 68, pp. 36–46, 2017.

[8] A. Martín, R. Lara-Cabrera, and D. Camacho, “Android malware detection through hybrid features fusion and ensemble classifiers: The AndroPyTool framework and the OmniDroid dataset,” Inf. Fusion, vol. 52, pp. 128–142, 2019.

[9] P. Feng, J. Ma, C. Sun, X. Xu, and Y. Ma, “A novel dynamic android malware detection system with ensemble learning,” IEEE Access, vol. 6, pp. 30996–31011, 2018.

[10] N. Potha, V. Kouliaridis, and G. Kambourakis, “An extrinsic random-based ensemble approach for android malware detection,” Connection Science, vol. 33, no. 4, pp. 1077–1093, 2020.

[11] R. M. Yadav, “Effective analysis of malware detection in cloud computing,” Comput. Secur., vol. 83, pp. 14–21, 2019.