مقاله پژوهشی

یک رویکرد کاهش ابعاد مبتنی بر یادگیری عمیق و الگوریتم کورکور بال سیاه برای تشخیص بدافزار اندروید

محسن اقبالی، محمّدرضا ملاخلیلی میبدی و کمال میرزائی

چکیده: امروزه با افزایش تعداد دستگاه‌های تلفن همراه، بدافزارهای مخرب برای پلتفرم اندروید نیز گسترش‌یافته‌اند. این بدافزارها با روش‌های پیچیده‌تری نوشته‌شده‌اند که تشخیص آنها دشوار است. برای تشخیص آنها از روش‌های یادگیری ماشین و یادگیری عمیق استفاده می‌شود زیرا توانایی تشخیص الگوهای پیچیده بدافزار را دارند. یکی از چالش‌های تشخیص بدافزار با روش‌های یادگیری ماشین و یادگیری عمیق، ابعاد زیاد نمونه‌های آموزشی است. در این مقاله برای کاهش دادن ابعاد نمونه‌های آموزشی در تشخیص بدافزارهای اندروید یک نسخه دودویی از الگوریتم کورکور بال سیاه ارایه می‌شود. در روش پیشنهادی در مرحله اول با الگوریتم کورکور بال سیاه، ویژگی‌های بدافزار استخراج می‌شود و این ویژگی‌ها تحویل شبکه عصبی LSTM می‌شود. نقش LSTM طبقه‌بندی نمونه‌ها به بدافزار و نرم‌افزار در اندروید است. برای افزایش دقت LSTM فراپارامترهای آن با استفاده از الگوریتم بهینه‌سازی محاسبات ریاضی نیز بهینه‌سازی می‌شود. آزمایش‌ها در مجموعه‌داده CICandMal۲۰۱۷ نشان‌داد دقت، حساسیت و صحت روش پیشنهادی به ترتیب ۹۸٫۶۳٪، ۹۸٫۲۹٪ و ۹۷٫۴۸٪ است. در رویکرد پیشنهادی اگر از متعادل‌سازی با روش GAN برای مجموعه‌داده CICandMal۲۰۱۷ استفاده‌شود آنگاه متوسط دقت، حساسیت و صحت روش پیشنهادی به مقادیر ۹۹٫۶۲٪، ۹۸٫۹۳٪ و ۹۸٫۵۲٪ افزایش‌داده می‌شود. آزمایش‌ها نشان‌می‌دهد که روش پیشنهادی نسبت به روش‌های کاهش ابعاد نظیر الگوریتم بهینه‌سازی وال، الگوریتم بهینه‌سازی شاهین و الگوریتم بهینه‌سازی کرکس دقت بیشتری در تشخیص بدافزار دارد. روش پیشنهادی نسبت به شبکه عصبی LSTM دقت بیشتری در حدود ٤٫١٦ درصد دارد.

کلیدواژه: **یادگیری ماشین، یادگیری عمیق، انتخاب ویژگی، کاهش ابعاد، الگوریتم بهینه‌سازی کورکور بال سیاه، بدافزار اندروید.**

۱- مقدمه

طبق گزارش‌های در سال ۲۰۲۱، محبوبیت تلفن‌های هوشمند در دهه گذشته با میلیاردها کاربر به سرعت افزایش‌یافته‌است. دلیل آن این است که تلفن‌های هوشمند بسیار مفید و راحت هستند. ارسال ایمیل، انجام بازی، گرفتن عکس و فیلم، جستجو در وب، استفاده از سیستم مکان‌یاب و موارد دیگر تنها بخشی از کاربردهای متعدد گوشی‌های هوشمند است.

این مقاله در تاریخ ۱۷ اردیبهشت ماه ۱۴۰۳ دریافت و در تاریخ ۷ آبان ماه ۱۴۰۴ بازنگری شد.

محسن اقبالی، گروه مهندسی کامپیوتر، واحد میبد، دانشگاه آزاد اسلامی، میبد، ایران، (email: mohsen.eghbali@iau.ir).

محمّدرضا ملاخلیلی میبدی (نویسنده مسئول)، گروه مهندسی کامپیوتر، واحد میبد، دانشگاه آزاد اسلامی، میبد، ایران، (mollakhalili@maybodiau.ac.ir).

کمال میرزائی، استادیار، گروه مهندسی کامپیوتر، واحد میبد، دانشگاه آزاد اسلامی، میبد، ایران، (email: kamal.mirzaie@iau.ac.ir).

برنامه‌های کاربردی روزانه در حال توسعه و بهبود هستند و دستیابی به این امر را ممکن‌می‌سازد، به‌ویژه در سیستم‌عامل اندروید که برای اولین بار به‌عنوان یک هسته لینوکس هک‌شده بهینه‌سازی‌شده برای گجت‌های موبایل با صفحه لمسی ظاهر شد [۱]. در چند سال گذشته برنامه‌های سیستم عامل اندروید به بیش از ۳ میلیون برنامه گسترش‌یافته‌است که نشان‌دهنده اهمیت این سیستم عامل است. بانکداری، رسانه‌های اجتماعی، مراقبت‌های بهداشتی، آموزش و سرگرمی تنها برخی از کاربردهای ممکن برای این برنامه‌های اندرویدی هستند. در نتیجه، بیشتر این برنامه‌ها برای نفع مصرف‌کنندگان نهایی خود استفاده می‌شوند. با این حال، برخی از آنها به طور مخرب توسط هکرها و بهره‌برداران استفاده می‌شوند. بدافزار به این برنامه‌های مضر اطلاق می‌شود که به عنوان نرم‌افزار تهاجمی تعریف می‌شود که داده‌ها را می‌دزدد یا به رایانه کاربر دیگر آسیب می‌رساند [۲]. مجرمان سایبری بدافزاری را ایجاد می‌کنند تا به طرق مختلف از جمله ابزارهای تبلیغاتی مزاحم، کرم‌ها، باج‌افزارها و ویروس‌های تروجان عمل کنند. از آنجایی که نرم‌افزارهای مخرب همیشه در حال توسعه هستند، خنثی‌کردن نقض‌های امنیتی به طور فزاینده‌ای چالش برانگیز است. به عنوان مثال، در سال ۲۰۲۱، یک موسسه امنیتی[1] به کاربران اندروید هشدار داد که میلیون‌ها تلفن هوشمند همراه در برابر بدافزار اسمیت[2] آسیب‌پذیر هستند. این جاسوس‌افزار همچنین از واتس‌اپ به عنوان پوششی برای حمله به سیستم‌های اندرویدی استفاده می‌کند [۳]. در سال ۲۰۲۱، گزارش‌شد که بیش از یک میلیارد گوشی هوشمند اندرویدی در معرض هک قراردارند، زیرا فاقد آخرین ارتقاء امنیتی هستند [۴]. علاوه بر این، کارشناسان آزمایشگاه کسپرسکی در سال ۲۰۲۰ دریافتند که هکرهای متعددی از فروشگاه برنامه گوگل[3] برای انتشار بدافزارهای پیچیده برای سال‌ها استفاده کرده‌اند [۵]. اخیراً بسیاری از حساب‌های فیس‌بوک با استفاده از برنامه بدافزار اندرویدی FlyTrap هک شدند [۶].

سیستم عامل اندروید دارای یک ماژول مجوز داخلی است که قبل از اعطای مجوزهای درخواست‌شده توسط یک برنامه اندروید، بررسی‌می‌کند که آیا یک خط مشی امنیتی، نقض شده است یا خیر. جامعه تحقیقاتی دو روش را برای شناسایی بدافزارها به کار می‌برند که عبارتند ار تجزیه و تحلیل بدافزار استاتیک[4]، پویا[5] و ترکیبی[6] می‌باشند. تجزیه و تحلیل

1. Cybersecurity Check Point
2. Agent Smith
3. Google Play
4. Static Malware Detection
5. Dynamic Malware Detection
6. Hybrid Malware Detection

استاتیک بدافزار جایی است که برنامه‌ها بدون اجرا بررسی می‌شوند، در حالی‌که تجزیه و تحلیل دینامیک رفتار بدافزار را در جعبه ایمنی پس از اجرا ارزیابی‌می‌کند. علی‌رغم نقش فناوری‌های کنونی در بهبود کیفیت زندگی و گسترش دنیای سایبری، تهدیدات سایبری به سطح جدیدی رسیده‌اند و با سرعت ترسناکی در حال افزایش هستند. مهمتر از آن، حملات جدیدی که می‌توانند دفاعی گوشی‌های هوشمند را نقض‌کنند، دائما در حال توسعه و انتشار هستند [۷]. مزیت اصلی استفاده از تکنیک‌های یادگیری ماشین[1] (ML) [۸] و یادگیری عمیق[2] (DL) [۹] برای تشخیص آسیب‌پذیری این است که می‌تواند تکنیک‌هایی را شناسایی‌کند که قبلاً دیده نشده‌اند. مزیت دیگر اتوماسیون فرآیند است، زیرا به عوامل خارجی بستگی ندارد. در نهایت، آنها توانایی واکنش سریع و شناسایی سریع آنها را دارند که این کیفیتی است که در یک سیستم امنیتی بسیار ارزشمند است [۱۰].

از سوی دیگر می‌توان از این فناوری برای ایجاد تهدیدات پیچیده‌تری نیز استفاده کرد. به نوبه خود، آنها همچنین می‌توانند از آن برای حمله به مدل و یافتن نقاط ضعف در آن استفاده‌کنند تا تهدیداتی را ایجادکنند که قابل‌شناسایی نیستند. یک روش استاندارد برای تشخیص بدافزار استفاده از ویژگی‌های استاتیک و پویاست. مدیریت این مجموعه داده‌های عظیم به دلیل پیچیدگی آنها کار آسانی نیست. این می‌تواند توانایی یادگیری را مختل‌کند یا حتی زمان را طولانی‌تر کند. روش‌های کاهش ویژگی برای کاهش ابعاد داده‌ها ضروری هستند، زیرا برخی از ویژگی‌ها در مجموعه داده‌ها غیرضروری و اضافی هستند [۱۱]. برای انتخاب ویژگی و کاهش ابعاد در تشخیص بدافزار اندروید تاکنون الگوریتم‌های مختلفی، ارایه‌شده که بیشتر آنها از نوع فراابتکاری از جمله الگوریتم ژنتیک[3] (GA) [۱۲] و الگوریتم بهینه‌سازی ذرات[4] (PSO) [۱۳] است.

محققان امنیتی کسپرسکی دریافتند که سهم باج‌افزاری که روزانه با آن مواجه می‌شود، ۱۸۱ درصد نسبت به سال ۲۰۲۱ افزایش یافته و به ۹۵۰۰ فایل رمزگذاری‌شده در روز رسیده است. همچنین کارشناسان امنیتی کسپرسکی افزایش ۱۰ درصدی در سهم فایل‌های مخربی را که پلتفرم اندروید را هدف قرار می‌دهند، شناسایی‌کردند. کمپین‌های بدنام ۲۰۲۲، هزاران کاربر اندروید را در سراسر جهان کمین کردند و نمونه‌های بارز این روند هستند. گزارش دفاعی باج‌افزار SpyCloud در سال ۲۰۲۲ بیش از ۳۰۰ نفر را در نقش‌های امنیتی IT فعال در سازمان‌های ایالات متحده، بریتانیا و کانادا مورد بررسی قرار داد و حداقل ۵۰۰ کارمند در سال ۲۰۲۲ خطر باج‌افزار را ارزیابی‌کردند. این نظرسنجی نشان داد که ۹۰٪ سازمان‌ها تحت تأثیر قرارگرفته‌اند. توسط باج‌افزار در سال ۲۰۲۲ که در مقایسه با سال ۲۰۲۱، که در آن درصد ۷۲٫۵٪ بود، افزایش قابل‌توجهی دارد. سیمانتک گزارش‌داد که بیش از ۵۰ درصد بدافزارهای جدید در واقع انواع بدافزارهای موجود هستند. علاوه بر این، موسسه AV-TEST هر روز بیش از ۴۵۰۰۰۰ برنامه مخرب جدید (بدافزار) و همچنین برنامه‌های ناخواسته بالقوه را شناسایی‌می‌کند [۱۴]. مهاجمان سایبری تلاش‌می‌کنند از آسیب‌پذیری‌های شناخته و ناشناخته برای ایجاد نفوذ موفق سوءاستفاده‌کنند. با استقرار تعداد گزاف اشیاء متصل، به راحتی مقیاس حملات بدافزاری را که می‌توان بر روی این دستگاه‌ها راه‌اندازی‌کرد، تصورکرد. این منجر به یک چالش بزرگ‌می‌شود که باید برطرف‌شود. تجزیه و تحلیل بدافزار، فرآیندی است برای بررسی فایل‌های اجرایی، با هدف استخراج هر چه بیشتر اطلاعات ارزشمند. هدف از تحلیل داده‌ها ترسیم محیط یک حمله سایبری و شناسایی عملکردها و رفتارهای مختلف برنامه مخرب است [۱۴].

هدف از این مقاله ارایه یک رویکرد جدید برای انتخاب ویژگی و کاهش ابعاد در مجموعه داده‌های اندروید برای تشخیص بدافزار با استفاده از الگوریتم بهینه‌سازی کورکور بال سیاه[5] [۱۵] است که در سال ۲۰۲۴ ارایه‌شده‌است. اهمیت این موضوع در آن است که بدافزارهای اندروید می‌توانند باعث سرقت اطلاعات کاربران شوند و اطلاعات شخصی آنها را لو دهند. زیان بدافزارهای اندروید بسیار قابل‌توجه است و از این جهت تشخیص بدافزار از اهمیت زیادی برخوردار است. نوآوری روش پیشنهادی در ارایه یک نسخه دودویی از الگوریتم کورکور بال سیاه برای کاهش ابعاد در تشخیص بدافزار اندروید است و از طرفی بهبود جستجوی اکتشافی این الگوریتم و ترکیب آن با روش درخت تصمیم‌گیری یک نوآوری روش پیشنهادی است. یک نوآوری دیگر روش پیشنهادی، بهینه‌سازی پارامترهای مرتبط با LSTM[6] با الگوریتم بهینه‌سازی محاسبات ریاضی[7] ریاضی[7] (AOA) [۱۶] است. سهم نویسندگان در توسعه یک روش تشخیص بدافزار اندروید به شرح ذیل است:

- ارایه یک نسخه دودویی از الگوریتم کورکور بال سیاه برای کاهش ابعاد در تشخیص بدافزار اندروید.
- بهینه‌سازی و بهبود جستجوی اکتشافی الگوریتم کورکور بال سیاه
- بهینه‌سازی ورودی‌ها LSTM با الگوریتم کورکور بال سیاه دودویی
- بهینه‌سازی فراپارامترهای LSTM با الگوریتم بهینه‌سازی محاسبات ریاضی
- تلفیق الگوریتم کورکور بال سیاه و درخت تصمیم‌گیری برای کاهش ابعاد
- تلفیق هوش گروهی و الگوریتم‌های فراابتکاری بر پایه ریاضی برای بهبود دقت روشهای یادگیری عمیق از جمله LSTM در تشخیص بدافزار اندروید

این مقاله در بخش ۲ کارهای مرتبط در زمینه تشخیص بدافزار اندروید را مرورمی‌کند. در بخش ۳، سیستم تشخیص بدافزار با هوش گروهی و یادگیری عمیق ارایه‌می‌شود. بخش ۴، به پیاده‌سازی روش پیشنهادی برای تشخیص بدافزار اندروید و مقایسه آن با روش‌های مشابه می‌پردازد. بخش ۵، شامل نتایج و یافته‌های تحقیق به همراه پیشنهادهای آتی برای بهبود روش پیشنهادی در تشخیص بدافزار اندروید است.

۲- کارهای مرتبط

در [۱۷]، یک چارچوب گروهی مبتنی بر یادگیری عمیق برای تشخیص قوی بدافزارهای اندروید ارایه دادند. این مطالعه یک رویکرد جدید تشخیص بدافزار را با استفاده از مجموعه‌ای از شبکه‌های عصبی کانولوشن[8] (CNN) برای افزایش دقت طبقه‌بندی معرفی‌می‌کند. این روش شامل یک فرآیند چندمرحله‌ای است که با استخراج و پیش‌پردازش

1. Machine Learning
2. Deep Learning
3. Genetic Algorithm
4. Particle Swarm Optimization
5. Black-Winged Kite Algorithm
6. Long Short-Term Memory
7. Arithmetic Optimization Algorithm
8. Convolutional Neural Network

فایل‌های برنامه اندروید APK[1] شروع می‌شود. مرحله پیش‌پردازش شامل خارج‌کردن از حالت فشرده، خارج‌کردن از حالت کامپایل و تبدیل فایل‌های APK به فایل‌های بایت‌کد و Dex است. داده‌های بایت استخراج شده به بردارهای یک بعدی تبدیل شده و به تصاویر دو بعدی خاکستری تغییر شکل می‌دهند و امکان یادگیری ویژگی کارآمد را از طریق CNNها فراهم می‌کنند. به طور خاص، این مدل در مجموعه داده Drebin به دقت ۹۸٫۶۵٪، امتیاز F1 برابر با ۹۶٫۴۳٪ و در مجموعه داده AMD به دقت ۹۷٫۹۱٪، امتیاز F1 برابر با ۹۶٫۷۳٪ دست می‌یابد.

در [۱۸]، یک روش تشخیص و طبقه‌بندی بدافزارهای اندرویدی ترکیبی با استفاده از شبکه‌های عصبی عمیق ارایه‌دادند. برخلاف رویکردهای قبلی، سیستم پیشنهادی، تجزیه و تحلیل چندبعدی از مجوزها، اهداف و فراخوانی‌های API اندروید را ادغام‌می‌کند و استخراج ویژگی قوی را حتی تحت محدودیت‌های مهندسی معکوس امکان‌پذیر می‌سازد. نتایج تجربی، عملکرد پیشرفته‌ای را نشان‌می‌دهند و به دقت ۹۸٫۲٪ (بهبود ۷٫۵٪ نسبت به DeepAMD) در ارزیابی مجموعه داده‌های متقابل شامل ۱۵ خانواده بدافزار و ۴۵۰۰۰ برنامه دست می‌یابند.

در [۱۹]، یک رویکرد کارآمد برای تشخیص بدافزار اندروید از طریق یادگیری عمیق ارایه‌دادند. برای ساده‌سازی تشخیص بدافزار، این مطالعه با تبدیل داده‌های ترافیک شبکه به تصاویر، رویکرد جدیدی را ارائه می‌دهد که سپس با استفاده از مدل‌های یادگیری عمیق تجزیه و تحلیل می‌شوند. آنها مدل‌های ترکیبی را معرفی می‌کنند که به طور یکپارچه شبکه‌های عصبی کانولوشن (CNN) و تبدیل‌کننده‌های بینایی[2] (ViT) را ادغام‌می‌کنند تا از نقاط قوت مربوطه خود در شناسایی ترافیک مخرب بهره‌ببرند. این تحقیق نه تنها استاندارد جدیدی را در کارایی تشخیص بدافزار اندروید تعیین‌می‌کند، بلکه راه را برای پیشرفت‌های آینده در کاربرد یادگیری عمیق برای امنیت سایبری هموارمی‌کند.

در [۲۰]، مدل یادگیری عمیق ترکیبی برای تشخیص دقیق و کارآمد بدافزارهای اندرویدی با استفاده از DBN-GRU را پیشنهاد دادند. این مطالعه یک مدل یادگیری عمیق ترکیبی (DBN-GRU) را معرفی‌می‌کند که شبکه‌های باور عمیق (DBN) را برای تحلیل استاتیک و واحدهای بازگشتی دروازه‌دار[3] (GRU) را برای مدل‌سازی رفتار پویا ادغام می‌کند تا تا دقت و کارایی تشخیص بدافزار را افزایش دهد. این مدل ویژگی‌های ایستا (مجوزها، فراخوانی‌های API، فیلترهای هدف) و پویا (فراخوانی‌های سیستم، فعالیت شبکه، ارتباطات بین فرآیندی) را از APKهای اندروید استخراج می‌کند و امکان تجزیه و تحلیل جامعی از رفتار برنامه را فراهم می‌کند. مدل پیشنهادی بر روی مجموعه داده Drebin آموزش و آزمایش شده‌است که شامل ۱۲۹۰۱۳ برنامه (۵۵۶۰ بدافزار و ۱۲۳۴۵۳ برنامه بی‌خطر) است. ارزیابی عملکرد در برابر NMLA-AMDCEF، MalVulDroid و LinRegDroid نشان‌داد که DBN-GRU به دقت ۹۸٫۷٪، صحت ۹۸٫۵٪، یادآوری ۹۸٫۹٪ و AUC ۰٫۹۹ دست یافته است که از مدل‌های مرسوم بهتر عمل‌می‌کند. علاوه بر این، زمان‌های پیش‌پردازش، استخراج ویژگی و طبقه‌بندی بدافزار سریع‌تری را نشان‌می‌دهد که آن را برای استقرار در زمان واقعی مناسب می‌کند. DBN-GRU با ایجاد پل بین روش‌های تشخیص ایستا و پویا، قابلیت‌های تشخیص بدافزار را افزایش می‌دهد و در عین حال مثبت‌های کاذب و سربار محاسباتی را کاهش می‌دهد.

در [۲۱]، یک روش ایستا برای تشخیص بدافزار اندروید مبتنی بر فراخوانی API ارایه دادند. در این مطالعه الگوریتم‌های یادگیری ماشین محبوب مانند جنگل تصادفی، ماشین بردار پشتیبان، نزدیکترین همسایه K، رگرسیون لجستیک و رگرسیون تقویت گرادیان برای طبقه‌بندی استفاده می‌شود. ارزیابی‌ها نشان می‌دهد دقت روش آنها به حدود ۹۸٪ می‌رسد.

در [۲۲]، یک روش تشخیص بدافزار اندروید با استفاده از شبکه‌های عصبی کانولوشن گراف مکرر ارایه دادند. فرآیند اصلی این روش شامل ساخت یک گراف آپکد دالویک، استخراج زیرگراف‌های مکرر و جاسازی زیرگراف‌ها با استفاده از شبکه‌های عصبی کانولوشن گراف برای استخراج ویژگی‌های توپولوژیکی و آموزش مدل‌های طبقه‌بندی است. به طور خاص، دقت تشخیص آن تقریباً ۹۵٪ است و هزینه زمان برای یک تشخیص واحد از ۰٫۱ ثانیه تجاوز نمی‌کند.

در [۱۰]، یک روش تشخیص بدافزار اندروید از طریق یادگیری عمیق با استفاده از مکانیزم رای‌گیری گروهی ارایه شده است. این مقاله با تلاش برای شناسایی انواع مبهم بدافزار اندروید، رویکردی را برای رسیدگی به چالش‌ها و مسائل مربوط به طبقه‌بندی و شناسایی انواع مبهم بدافزار پیشنهاد می‌کند. طرح تشخیص و طبقه‌بندی به‌کار گرفته‌شده از تحلیل استاتیک و پویا با استفاده از مکانیزم رای‌گیری گروهی استفاده می‌کند. این مطالعه نشان می‌دهد که زیرمجموعه کوچکی از ویژگی‌ها زمانی که از بدافزار اصلی (غیر مبهم) مشتق شده‌اند، به‌طور مداوم خوب عمل می‌کنند، با این حال، پس از اعمال یک رویکرد مبهم‌سازی مبتنی بر ویژگی جدید، این مطالعه تغییر شدیدی را نشان می‌دهد که نشان‌دهنده اهمیت نسبی است. یکی از این ویژگی‌ها در مبهم‌کردن برنامه‌های بدخیم و بدافزار است. آزمایش‌ها نشان‌می‌دهد که مدل پیشنهادی بدافزار را به‌طور مؤثر و دقیق همراه با شناسایی ویژگی‌هایی که معمولاً توسط مهاجمان بدافزار مبهم می‌شوند، شناسایی می‌کند.

در [۲۳]، تشخیص بدافزار اندروید بر اساس جنگل عمیق را پیشنهاد دادند. در این مقاله، آنها یک چارچوب تشخیص دومرحله‌ای را بر اساس افزایش ویژگی و جنگل عمیق آبشاری پیشنهاد دادند. این روش می‌تواند ترافیک ایجادشده در فرآیند انتقال رمزگذاری‌شده بدافزار اندروید را شناسایی کند. مرحله اول طبقه بندی دودویی نرم‌افزارهای بدخیم و بدخیم را درک می‌کند. مرحله دوم چندطبقه‌بندی دسته‌های مختلف بدافزار را متوجه می‌شود. برای افزایش نمایش داده‌ها، شبکه‌های عصبی کانولوشن برای استخراج ویژگی‌های بدخیم در مرحله اول و از روش تحلیل مؤلفه اصلی برای استخراج ویژگی‌های مخرب در مرحله دوم استفاده می‌شود. این ویژگی‌های استخراج‌شده با بخش محموله ترافیک ترکیب می‌شوند تا ویژگی‌های همجوشی را برای کار طبقه‌بندی تشکیل دهند. به منظور انطباق با مقیاس‌های مختلف نمونه‌ها، به ویژه برای نمونه در مقیاس کوچک، روش جنگل عمیق آبشاری برای ساخت مدل طبقه‌بندی پیشنهاد شده است. در این مدل، بسیاری از لایه‌ها که از طبقه‌بندی‌کننده‌های پایه تشکیل شده‌اند، به صورت آبشاری هستند و می‌توان تعداد لایه‌ها را به‌طور خودکار با توجه به مقیاس نمونه‌ها تنظیم کرد. نتایج تجربی بر روی چندین مجموعه‌داده ثابت می‌کند که روش پیشنهادی برای تشخیص انتقال رمزگذاری‌شده بدافزار اندروید موثر است و همچنین برای تشخیص حملات ناشناخته مناسب است.

در [۲۴]، مدل‌های یادگیری عمیق برای شناسایی بدافزار در برنامه‌های اندروید بررسی‌شده‌است. علیرغم افزایش استفاده از برنامه‌های اندروید و

1. Android Package Kit
2. Vision Transformer
3. Gated Recurrent Units

حملات سایبری، استفاده از مدل‌های یادگیری عمیق برای شناسایی بدافزارهای نوظهور در برنامه‌های اندروید هنوز در حال ظهور است. بنابراین، این بررسی به دنبال توضیح مدل‌های یادگیری عمیق است که برای شناسایی بدافزارها در برنامه‌های اندرویدی استفاده می‌شوند، عملکرد آنها را بررسی کرده و همچنین شکاف‌های تحقیقاتی در حال ظهور را شناسایی کرده و توصیه‌هایی را برای کارهای آینده ارائه می‌کند. این مطالعه نشان‌داد که شبکه‌های عصبی کانولوشن، شبکه‌های عصبی بازگشتی دروازه‌ای، شبکه‌های عصبی عمیق، حافظه کوتاه‌مدت دوطرفه، حافظه کوتاه‌مدت بلندمدت (LSTM) و برجسته‌ترین مدل‌های تشخیص نرم‌افزار مخرب مبتنی بر یادگیری عمیق در اندروید هستند. یافته‌ها نشان‌می‌دهد که مدل‌های یادگیری عمیق به طور فزاینده‌ای به یک تکنیک مؤثر برای تشخیص نرم‌افزارهای مخرب در برنامه‌های اندروید در زمان واقعی تبدیل می‌شوند.

در [۲۵]، یک روش تشخیص بدافزار اندروید بر اساس یادگیری عمیق چندوجهی و تجزیه و تحلیل ترکیبی ارایه دادند. روش‌های تشخیص بدافزار یادگیری ماشین سنتی بر مهندسی ویژگی‌های دستی تکیه می‌کنند که به دانش متخصص نیاز دارد. از سوی دیگر، روش‌های تشخیص بدافزار یادگیری عمیق استخراج خودکار ویژگی‌ها را انجام می‌دهند، اما معمولاً به داده‌ها و قدرت پردازش بسیار بیشتری نیاز دارند. در این کار، آنها یک روش جدید تشخیص بدافزار اندروید یادگیری عمیق چندوجهی، را پیشنهاد می‌کنند که مهندسی ویژگی‌های دستی و خودکار را با استفاده از شبکه‌های عصبی کانولوشن، شبکه‌های عصبی عمیق و شبکه‌های ترانسفورماتور ترکیب می‌کند. آنها از مجموعه داده‌های معیار Android Omnidroid در دسترس عموم استفاده کردند که حاوی داده‌های تحلیل استاتیک و پویا است که از ۲۲۰۰۰ نمونه بدافزار واقعی و نرم‌افزار خوب استخراج شده است. ارزیابی‌ها نشان‌داد دقت روش پیشنهادی از روش‌های DNN، CNN و TN در شناسایی بدافزار اندروید بیشتر است.

در [۲۶]، یک روش تشخیص بدافزار اندروید بر اساس شبکه‌های توجه گراف و ادغام عمیق ویژگی‌های چندوجهی ارایه‌شده‌است. نتایج تجربی نشان‌می‌دهد که روش آنها به دقت ۹۷٫۲۸٪ - ۹۹٫۵۴٪ در سه مجموعه‌داده ساخته‌شده دست می یابد که بهتر از روش‌های موجود است.

در [۲۷]، یک روش تشخیص بدافزار با الگوریتم بیوه سیاه و یادگیری عمیق ارایه‌شده‌است. هدف اصلی تکنیک پیشنهادی در طبقه‌بندی خودکار بدافزار اندروید نهفته است. برای انجام این کار، رویکرد انتخاب ویژگی مبتنی بر الگوریتم بیوه سیاه برای افزایش عملکرد طبقه‌بندی می‌شود. برای اهداف طبقه‌بندی بدافزار اندروید، تکنیک پیشنهادی از یک مدل ماشین یادگیری افراطی عمیق استفاده می‌کند و پارامتر آن به‌طور بهینه توسط الگوریتم بهینه‌سازی شیر مورچه انتخاب می‌شود. شبیه‌سازی تکنیک پیشنهادی بر روی مجموعه داده CICandMal۲۰۱۷ آزمایش‌شده‌است. نتایج آزمایشی گسترده نشان‌دهنده عملکرد بهتر تکنیک پیشنهادی نسبت به سایر آشکارسازهای بدافزار با حداکثر دقت ۹۸٫۹۵٪ است.

در [۲۸]، یک روش یادگیری رفتارهای مبتنی بر جریان و نمودار برای تشخیص بدافزار اندروید ارایه دادند. این مطالعه یک روش پیشنهادی بر اساس LSTM دوطرفه[1] (BiLSTM) و یک شبکه عصبی گراف[2] (GNN) به عنوان زیرشبکه ارایه شده است. آنها ویژگی‌های کد موقت استفاده‌شده توسط زیرشبکه BiLSTM از ردیابی تماس استخراج می‌شوند، در حالی که ویژگی‌های نمودار جریان مورد استفاده توسط زیرشبکه GNN از تمام ردیابی‌های تماس و ارتباطات بین مؤلفه‌ای ساخته می‌شوند. نتایج تجربی بر روی بیش از ۱۸۰۰۰ برنامه دنیای واقعی و بدافزار رایج نشان می‌دهد که روش آنها به بهبود قابل توجهی دست می‌یابد.

در [۲۹]، یک رویکرد فرا طبقه‌بندی‌کننده یادگیری عمیق بر پایه شبکه EfficientNet برای تشخیص بدافزار اندرویدی مبتنی بر تصویر ارایه دادند. این کار از ۲۶ مدل از پیش آموزش‌دیده مبتنی بر CNN استفاده می‌کند و بررسی و تجزیه و تحلیل دقیق آزمایش‌ها بر روی مجموعه‌داده بدافزار اندروید مبتنی بر تصویر نشان داده شده است. ویژگی‌های لایه ماقبل آخر مدل‌های پیش‌آموزش‌شده مبتنی بر کانولوشن استخراج می‌شوند و ابعاد ویژگی‌ها با استفاده از تحلیل مؤلفه اصلی هسته کاهش می‌یابد. ویژگی‌های کاهش‌یافته با هم ادغام شدند و برای طبقه‌بندی به یک متا طبقه‌بندی‌کننده یا طبقه‌بندی‌کننده انباشته منتقل شدند. این طبقه‌بندی دارای دو سطح است. در سطح اول ماشین بردار پشتیبان و طبقه‌بندی‌کننده یادگیری ماشین جنگل تصادفی برای پیش‌بینی و رگرسیون لجستیک در سطح دوم برای طبقه‌بندی قرار گرفتند. آزمایش‌ها نشان‌داد روش آنها از روش‌های یادگیری عمیق مانند DenseNet، ResNet، InceptionResNet دقت بیشتری دارد.

در [۳۰]، یک روش تشخیص بدافزار مبتنی بر یادگیری عمیق برای اندروید ارایه دادند. در این پژوهش یک معماری یادگیری عمیق برای شناسایی برنامه‌های بدافزار اندروید بر اساس ویژگی‌های ایستا و پویا پیشنهاد شده است. روش پیشنهادی از دو مدل تشخیص اصلی تشکیل شده است که اولی از روش یادگیری عمیق CNN-BiLSTM برای شناسایی بدافزار از تجزیه و تحلیل استاتیک استفاده‌می‌کند. مدل دیگر از Autoencoders عمیق به عنوان یک مدل تشخیص ناهنجاری برای شناسایی بدافزار بر اساس تجزیه و تحلیل پویا استفاده می‌کند. عملکرد معماری روش آنها با استفاده از دو مجموعه‌داده مختلف ارزیابی می‌شود. نتایج نشان می‌دهد که از روش CNN-BiLSTM و Deep Autoencoders دقت بیشتری دارد.

در [۳۱]، یک روش طبقه‌بندی و شناسایی بدافزار اندروید با استفاده از تجزیه و تحلیل ویژگی‌های URL ارایه دادند. برای استخراج ویژگی‌های ترتیب‌یافته، از روش تحلیل N-gram استفاده می‌شود و پس از آن، از روش تجزیه ارزش منفرد برای کاهش ویژگی‌ها با حفظ معنای واقعی استفاده می‌شود. ویژگی‌های نهفته با استفاده از ابزار تحلیل معنایی پنهان استخراج‌می‌شوند. در نهایت، CNN-LSTM، یک رویکرد یادگیری عمیق، برای طبقه‌بندی و شناسایی موثر بدافزار طراحی‌شده‌است.

در [۳۲]، یک روش تشخیص بدافزار اندرویدی مبتنی بر یادگیری عمیق ارایه دادند. در این پژوهش یک روش تشخیص بدافزار مبتنی بر یادگیری و طبقه‌بندی خانواده[3] (DeepMDFC) را برای شناسایی و طبقه‌بندی برنامه‌های مخرب اندروید از طریق تجزیه و تحلیل استاتیک و شبکه‌های عصبی مصنوعی عمیق پیشنهاد می‌کند. یافته‌های تجربی نشان می‌دهد که این روش از الگوریتم‌های یادگیری ماشین استاندارد پیشی‌می‌گیرد. در جدول ۱، یک دسته‌بندی از کارهای مرتبط نشان داده شده است.

1. Bidirectional Long Short-Term Memory
2. Graph Neural Network
3. Malware Detection and Family Classification Method

جدول ۱: دسته‌بندی از کارهای مرتبط در تشخیص بدافزار.

مرجع	رویکرد	دسته	مزیت	چالش
[۱۰]	رای‌گیری گروهی	تحلیل استاتیک و پویا	تشخیص بدافزار مبهم	پیچیدگی و زمان اجرای بالا
[۱۷]	CNN گروهی	یادگیری عمیق	دقت بالا	عدم تعادل مجموعه داده
[۱۸]	شبکه‌های عصبی عمیق	ترکیبی (پویا و استاتیک)	دقت بالا	عدم انتخاب ویژگی هوشمندانه
[۱۹]	ViT+CNN	یادگیری عمیق	دقت بیشتر از CNN	پیچیدگی زیاد
[۲۰]	DBN-GRU	یادگیری عمیق و ترکیب ایستا و پویا	دقت بیشتر از GRU	پیچیدگی بیشتر از GRU و DBN
[۲۱]	تحلیل با روشهای یادگیری ماشین	تحلیل ایستا	دقت در حدود ۹۸٪	عدم انتخاب ویژگی
[۲۲]	شبکه عصبی کانولوشن گراف مکرر	تحلیل ایستا و یادگیری عمیق	استخراج ویژگی‌های توپولوژیکی	مصرف حافظه زیاد
[۲۳]	جنگل عمیق	تحلیل استاتیک و یادگیری عمیق	دقت بالا	پیچیدگی بالا
[۲۴]	شبکه‌های عصبی عمیق	تحلیل استاتیک و شبکه عصبی عمیق	تشخیص ویژگی‌های مکانی و زمانی	عدم تعادل در مجموعه داده
[۲۵]	یادگیری عمیق چندوجهی	تحلیل استاتیک و پویا	دقت بیشتر از DNN، CNN و TN	عدم بهینه‌سازی فراپارامترهای مدل
[۲۶]	شبکه‌های توجه گراف	تحلیل استاتیک	دقت ۹۷.۲۸٪ – ۹۹.۵۴٪	پیچیدگی بالا
[۲۷]	الگوریتم بیوه سیاه و یادگیری عمیق	تحلیل استاتیک و بهینه‌سازی	دقت بالا	پیچیدگی بالا
[۲۸]	BiLSTM+GCN	تحلیل استاتیک و بهینه‌سازی	دقت بیشتر از LSTM	عدم تعادل مجموعه داده
[۲۹]	EfficientNet	تحلیل استاتیک و یادگیری عمیق	دقت بیشتر از ResNet	عدم آموزش روی تصاویر بدافزار

بررسی کارهای مرتبط نشان‌می‌دهد تکنیک‌های مختلفی برای تشخیص بدافزار استفاده‌می‌شود، از جمله رویکردهای مبتنی بر امضاء (استفاده از عبارات منظم، نام فایل‌ها، و غیره)، روش‌های مبتنی بر رفتار یا اکتشافی، و تجزیه و تحلیل ویژگی استاتیک و پویا می‌باشد. با این حال، این تکنیک‌ها در برابر انواع بدافزار جدید که از تکنیک‌های پنهان‌سازی پویا مانند مبهم‌سازی، چندشکلی، دگرگونی و بسته‌بندی استفاده‌می‌کنند، بی‌اثر شده‌اند. تجزیه و تحلیل پویا از تحلیل استاتیک خاص‌تر است زیرا هدف اصلی آن دستیابی به پوشش کد بالا برای نظارت بر هرگونه رفتار مخرب احتمالی است. این تکنیک‌ها بدون معایب نیستند، چالش اصلی میزان منابع مورد نیاز برای پیاده‌سازی کافی آن‌ها است که کاربرد آن‌ها را در دستگاه‌های دارای منابع محدود مانند تلفن‌های همراه محدودمی‌کند. علاوه بر این، بدافزارهای پیشرفته به دنبال شناسایی شبیه‌سازها یا سایر چارچوب‌های مورد استفاده برای تجزیه و تحلیل پویا هستند تا رفتار آنها را پنهان‌کرده و از تشخیص فرارکنند. تکنیک‌های ترکیبی وجود دارند که سودمندترین ویژگی‌های تحلیل استاتیک و دینامیکی را در خود جای‌می‌دهند. با این حال، این تکنیک‌ها میزان منابع مورد نیاز را کاهش‌نمی‌دهند و اغلب منابع سیستم را مصرف‌می‌کنند و در نتیجه زمان تجزیه و تحلیل بیش از حد در دستگاه‌های اندرویدی را به‌دنبال‌دارد. بررسی مطالعات نشان می‌دهد که در بیشتر موارد مجموعه داده‌ها برای تشخیص بدافزار نامتعادل می‌باشند و تعداد نمونه‌های کلاس خوش‌خیم از بدخیم بیشتر است و این عامل دقت یادگیری را کاهش می‌دهد. یک روش برای حل این مشکل که در روش پیشنهادی استفاده می‌شود، تولید نمونه‌های مصنوعی بدخیم و اضافه‌کردن به کلاس اقلیت با شبکه عصبی متخاصم است. یکی از روش‌های تشخیص بدافزار، روش‌های نظیر CNN یا LSTM است و این روش‌ها زمانی دارای دقت بالا هستند که فراپارامترهای آنها با دقت بهینه‌شوند و از این جهت برای رفع این چالش در روش پیشنهادی فراپارامترهای مدل بهینه‌سازی می‌شوند. در برخی از مطالعات از الگوریتم‌های آماری مانند کاهش مولفه اساسی یا الگوریتم‌های فراابتکاری فاقد هوش گروهی استفاده شده است اما در روش پیشنهادی برای انتخاب ویژگی یک رویکرد هوش گروهی و هوشمندانه‌تر برای کاهش ابعاد ورودی طبقه‌بندی‌کننده استفاده می‌شود.

۳- روش پیشنهادی

با افزایش محبوبیت و استفاده از سیستم‌عامل‌های اندروید، برنامه‌های مخرب با استفاده از روش‌ها و تکنیک‌های خلاقانه مورد هدف قرارمی‌گیرند. امروزه بدافزاری هوشمندمی‌شود که از روش‌های مختلفی از تکنیک‌های مبهم‌سازی برای پنهان‌کردن عملکرد خود و فرار از موتورهای ضد بدافزار استفاده می‌کند. افزایش سریع ضریب نفوذ گوشی‌های هوشمند در نتیجه منجر به افزایش حملات سایبری شده‌است. در این مقاله برای تشخیص بدافزار اندروید یک روش جدید بر پایه یادگیری عمیق و کاهش ابعاد با هوش گروهی ارایه شده است. در روش پیشنهادی در مرحله اول از الگوریتم بهینه‌سازی کورکور بال سیاه برای انتخاب ویژگی استفاده‌می‌شود و در مرحله دوم بعد از کاهش ابعاد توسط الگوریتم کورکور بال سیاه می‌توان از LSTM برای تشخیص بدافزار استفاده نمود. برای کاهش‌دادن خطای تشخیص بدافزار توسط LSTM پارامترهای آن توسط الگوریتم AOA بهینه‌سازی‌می‌شود.

روش پیشنهادی برای تشخیص بدافزار بدخیم و خوش‌خیم دارای مراحل ذیل است:

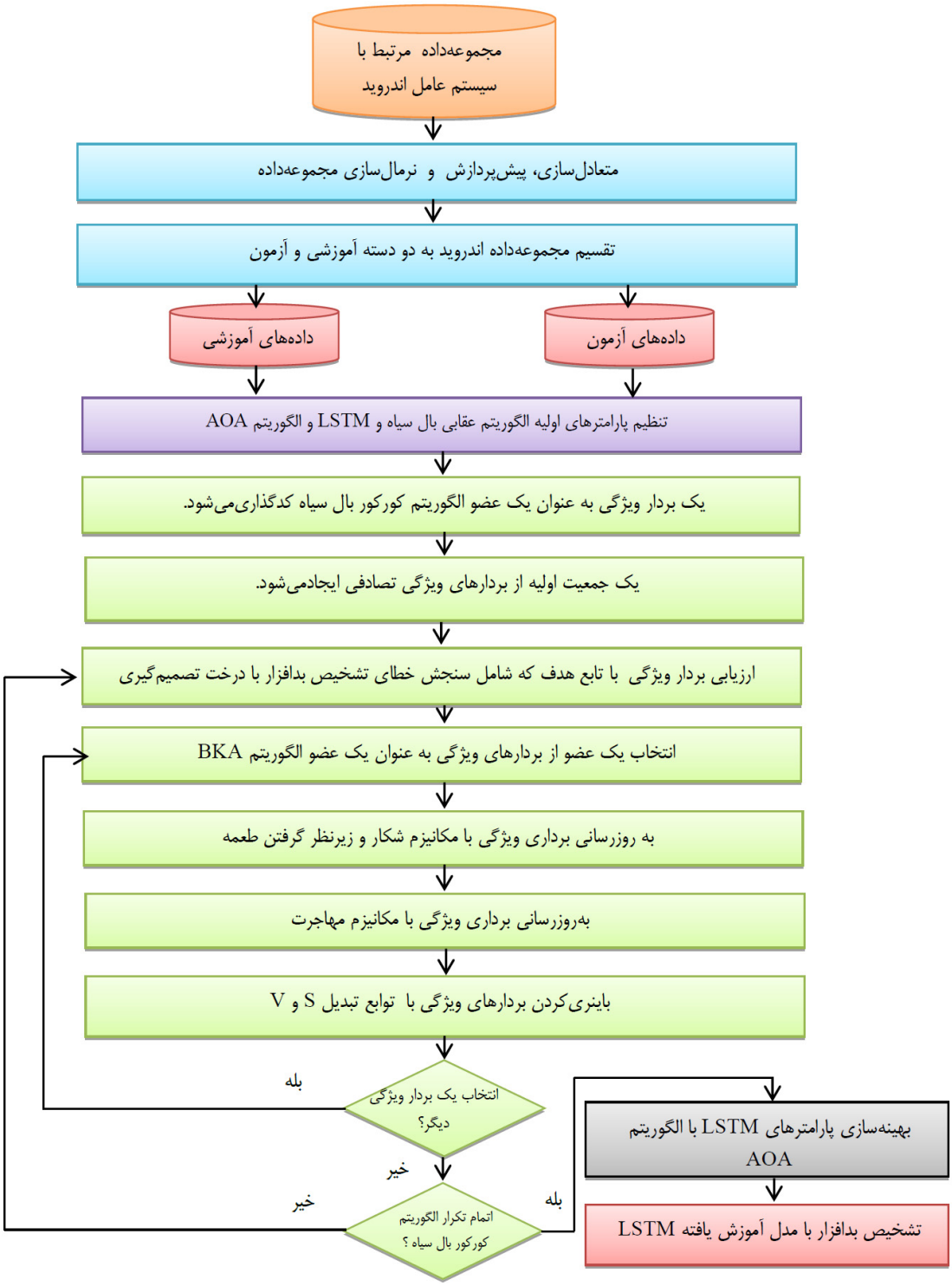
- متعادل‌سازی مجموعه‌داده با شبکه عصبی متخاصم[1] (GANs)
- پیش‌پردازش مجموعه‌داده نظیر نرمال‌سازی نمونه‌ها
- انتخاب ویژگی با الگوریتم کورکور بال سیاه و درخت تصمیم‌گیری
- بهینه‌سازی فراپارامترهای مدل LSTM با الگوریتم AOA

۳-۱ چارچوب پیشنهادی

در شکل ۱ چارچوب روش پیشنهادی برای کاهش ابعاد و تشخیص بدافزار اندروید با استفاده از الگوریتم کورکور بال سیاه ارایه شده است و مراحل ذیل در روش پیشنهادی به‌کار گرفته شده است:

- مجموعه‌داده به عنوان ورودی در نظر گرفته می‌شود و پیش‌پردازش می‌گردد.
- مجموعه‌داده مورد نرمال‌سازی می‌گردد.
- مجموعه‌داده با روش‌های مانند GAN متعادل‌سازی‌می‌شود تا کلاس اقلیت دارای نمونه‌های به اندازه کلاس اکثریت شود. در این حالت روش GAN تعدادی نمونه بدخیم مصنوعی تولید می‌کند و آنها را به کلاس اقلیت (کلاس بدخیم) اضافه می‌کند تا مجموعه داده متعادل‌سازی شود. مزیت دیگر GAN آن است نمونه‌های بدخیم

1. Generative Adversarial Networks



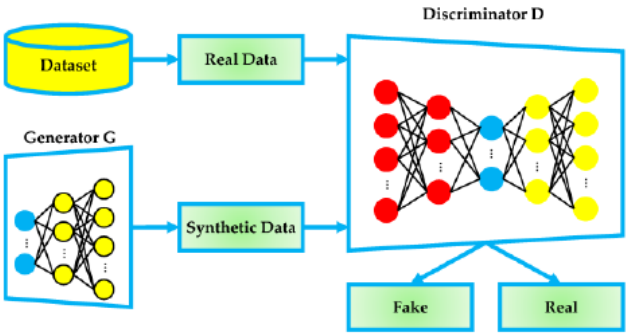
شکل ۱: چارچوب روش پیشنهادی برای تشخیص بدافزار آندروید.

جعلی و ناشناخته را به مجموعه‌داده اضافه می‌کند تا روش پیشنهادی توانایی بیشتری در تشخیص بدافزارهای ناشناخته داشته باشد.

- یک نسخه بهبودیافته از الگوریتم بهینه‌سازی کورکور بال سیاه برای کاهش ابعاد و انتخاب ویژگی ارایه می‌شود.
- پارامترهای الگوریتم LSTM با استفاده از الگوریتم AOA بهینه‌سازی می‌شود و از این مدل برای تشخیص بدافزار استفاده می‌گردد.
- مدل مورد آموزش برای تشخیص بدافزار با داده‌های آزمون ارزیابی و تحلیل می‌شود.

۳-۲ پیش‌پردازش و نرمال‌سازی

در این مقاله از MinMaxScaler برای نرمال‌سازی داده‌های عددی و از یک رمزگذار وان-هات در [۳۵]، برای پردازش ویژگی‌های دسته‌بندی‌شده برای ایجاد سازگاری در تمام مجموعه‌داده‌ها استفاده شده است. MinMaxScaler مقادیر عددی را در یک محدوده یکنواخت، بین ۰ و ۱، تنظیم و تضمین می‌کند که همه ویژگی‌ها مقیاس‌های قابل مقایسه‌ای دارند. این امر مانع از تسلط ویژگی‌هایی با مقدار بزرگتر بر فرآیند یادگیری می‌شود و تفاوت‌های نسبی آنها را حفظ می‌کند.



شکل ۲: تولید نمونه‌های بدخیم مصنوعی توسط روش GAN [۳۷].

رمزگذار وان- هات برای تبدیل متغیرهای دسته‌بندی‌شده به فرمت دودویی استفاده‌می‌شود که هر دسته با یک ستون مجزا نمایش داده‌می‌شود. این کدگذاری از ایجاد هرگونه رابطه ترتیبی بین دسته‌ها جلوگیری‌می‌کند و تضمین‌می‌کند که مدل با هر دسته به طور یکسان متمایز رفتارکند. با اعمال MinMaxScaler و کدگذاری وان- هات، به مدل اجازه‌داده‌می‌شود تا داده‌ها را به طور مؤثر تفسیر و از آنها یاد بگیرد [۳۶].

۳-۳ متعادل‌سازی مجموعه‌داده

شبکه‌های عصبی GAN به عنوان ابزاری قدرتمند در تشخیص بدافزار، به ویژه در ایجاد داده‌های مصنوعی برای آموزش مدل، عمل‌می‌کنند. شبکه‌های عصبی GAN مطابق شکل ۲، شامل دو شبکه عصبی، یک مولد و یک متمایزکننده هستند که به طور همزمان آموزش داده می‌شوند [۳۵]. مولد داده‌های مصنوعی تولیدمی‌کند، در حالی که متمایزکننده تلاش‌می‌کند تا بین داده‌های معتبر و مصنوعی تمایز قائل‌شود. شبکه‌های GAN می‌توانند نمونه‌های بدافزار مصنوعی تولیدکنند و به مشکل داده‌های برچسب‌گذاری شده ناکافی بپردازند. با این وجود، GANها می‌توانند از نظر محاسباتی پیچیده باشند و برای جلوگیری از تولید داده‌های غیرقابل قبول که ممکن است بر دقت تشخیص تأثیر منفی بگذارند.

هدف آموزش، رسیدن به تعادل است که در آن مولد، داده‌هایی چنان واقع‌گرایانه تولیدمی‌کند که متمایزکننده نمی‌تواند آن را به طور قابل اعتمادی از داده‌های واقعی تشخیص دهد. از نظر ریاضی، هدف یادگیری تخاصمی بین مولد و متمایزکننده به صورت (۱) بیان می‌شود [۳۶]:

$$\min_G \max_D V(D,G) = \mathbb{E}_{X\sim p_{data}(x)}[\log(D(X))] + \mathbb{E}_{Z\sim p_Z(Z)}[\log(1-D(G(Z)))] \quad (1)$$

که در آن $p_{data}(x)$ نشان‌دهنده توزیع داده‌های واقعی است که نمونه‌های واقعی $\boldsymbol{x}$ از آن استخراج می‌شوند، $p_z(Z)$ توزیع پیشین در فضای پنهان است که معمولاً به عنوان یک توزیع گاوسی یا یکنواخت چند متغیره انتخاب می‌شود، $G(Z)$ داده‌های مصنوعی تولیدشده از نویز Z است، و $D(\cdot)$ نشان‌دهنده احتمال تخمینی متمایزکننده مبنی بر واقعی بودن یک نمونه ورودی است. مولد G قصد دارد این تابع مقدار را با تولید نمونه‌هایی که $D(G(Z))$ را به حداکثر می‌رسانند، به حداقل برساند، بنابراین متمایزکننده را "فریب" می‌دهد تا نمونه‌های جعلی را به اشتباه به عنوان واقعی، طبقه‌بندی کند. در مقابل، متمایزکننده D سعی می‌کند با شناسایی صحیح ورودی‌های واقعی و مصنوعی، این هدف را به حداکثر برساند.

۳-۴ کاهش ابعاد

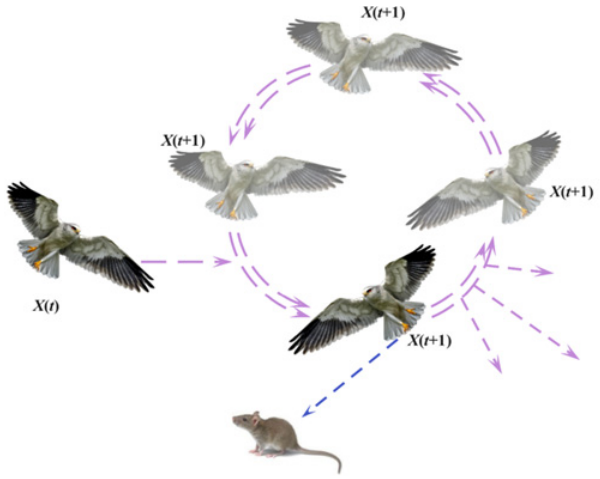
برای کاهش‌دادن ابعاد نمونه‌های مجموعه داده اندروید از الگوریتم بهینه‌سازی کورکور بال سیاه استفاده می‌شود. الگوریتم بهینه‌سازی کورکور بال سیاه دارای چهار معادله اصلی است که دو مورد از آنها برای شکار و دو مورد برای مهاجرت است. در معادلات شکار، الگوریتم سعی می‌کند جستجوی هوشمندانه محلی پیرامون جواب بهینه را انجام‌دهد. در رفتار مهاجرت الگوریتم تلاش‌دارد تا جمعیت را از نواحی که امید زیادی برای رسیدن به بهینگی نیست به مناطق بهینه سوق دهد. رفتارهای هوشمندانه و استراتژی شکار، حمله به طعمه، زیر نظر گرفتن طعمه و مهاجرت باعث می‌شود تا الگوریتم فضای مسئله را مورد پیمایش هوشمندانه قرار دهد. الگوریتم بهینه‌سازی کورکور بال سیاه بر خلاف روش‌های آماری مانند ضریب همبستگی و کاهش مولفه اساسی توانایی بالا و هوشمندانه‌ای برای تشخیص ویژگی‌های بهینه دارد. الگوریتم بهینه‌سازی کورکور بال سیاه بر خلاف RF و SVM نیازی به آموزش ندارد تا ویژگی‌های بهینه را پیدا نماید و از طرفی در زمان دوجمله‌ای الگوریتم‌های فرابتکاری می توانند راه‌های‌حل بهینه یا نزدیک بهینه را پیدا نمایند. در روش پیشنهادی هر بردار ویژگی یک عقاب و یک عضو الگوریتم بهینه‌سازی کورکور بال سیاه درنظرگرفته می‌شود. هر عقاب یا بردار ویژگی دارای مولفه‌های صفر و یک است و اگر هر مولفه صفر باشد نشان‌می‌دهد ویژگی انتخاب نشده و اگر یک مولفه برابر یک باشد آنگاه ویژگی برابر یک است. در روش پیشنهادی برای ارزیابی بردارهای ویژگی از تابع هدف (۲)، استفاده می شود.

$$\mathrm{Cost}(X_i) = aE(X_i) + b\frac{\|X_i\|}{Dim} \quad (2)$$

در این معادله، $E(X_i)$ خطای تشخیص بدافزار است و برای محاسبه آن بردار ویژگی به عنوان ورودی یک طبقه‌بندی‌کننده تعریف می‌شود تا مشاهده گردد که خطای تشخیص حملات و بدافزار به ازای این بردار ویژگی چقدر است. با توجه به اینکه روش درخت تصمیم‌گیری یک طبقه‌بندی‌کننده سبک و دقیق است از درخت تصمیم‌گیری برای ارزیابی بردارهای ویژگی و محاسبه E استفاده می‌شود. $\|X_i\|$ تعداد ویژگی انتخاب شده در بردار ویژگی X_i می‌باشد. Dim Dim ابعاد به‌کار رفته در مجموعه‌داده است. a و b b دو پارامتر تصادفی بوده که مجموع آنها برابر یک است. بردار ویژگی که این تابع را کمینه نماید به عنوان بردار ویژگی بهینه درنظر گرفته می‌شود. برای ارزیابی بردار ویژگی از درخت تصمیم‌گیری برای محاسبه خطای تشخیص بدافزار استفاده می‌شود. درخت تصمیم‌گیری از معیارها برای تعیین بهترین تقسیم ویژگی استفاده می‌کند، با روش‌های حریصانه که به طور سیستماتیک برای همه نقاط در یک منطقه تصمیم‌گیری اعمال‌می‌شود و در نتیجه مقادیر متریک بهتری به‌دست‌می‌آید. آنتروپی تصادفی بودن اطلاعات را اندازه‌گیری‌می‌کند و بر پیچیدگی تصمیم‌گیری تأثیرمی‌گذارد. هدف به حداقل رساندن آنتروپی برای مناطق تصمیم‌گیری همگن است. آنتروپی (E) از طریق (۳) محاسبه می‌شود.

$$E = -\sum_{1}^{N} P_i \log_2 P_i \quad (3)$$

که در آن P برابر احتمال یک کلاس در مجموعه‌داده است. شاخص جینی احتمال خطای طبقه‌بندی را با استفاده از (۴) اندازه‌گیری می‌کند.



شکل ۳: رفتار جستجو و کاوش در الگوریتم کورکور بال سیاه [۱۵].

$$GiniIndex = GI = \sum_{i=1}^{C} p_i(1-p_i) \quad (۴)$$

که در آن P برابر احتمال طبقه‌بندی نمونه است. متریک افزایش اطلاعات طبقه‌بندی مربوط به GI و آنتروپی را تصحیح‌می‌کند. فرآیند تقسیم با استفاده از آنتروپی یا شاخص جینی باعث کاهش خطا می‌شود. بهره اطلاعات توسط (۵) به‌دست می‌آید.

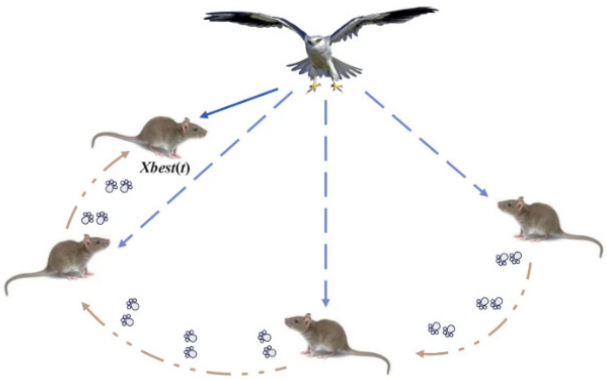
$$Info\_Gain = IG = E_{parent} - E_{children} \quad (۵)$$

که در آن E_{parent} برابر آنتروپی قبل از تقسیم و $E_{children}$ برابر آنتروپی پس از تقسیم است. کورکور بال سیاه پرنده‌ای کوچک با بالاتنه خاکستری آبی و پایین تنه سفید است. ویژگی‌های قابل توجه آنها شامل مهاجرت و رفتار شکار است. آنها از پستانداران کوچک، خزندگان، پرندگان و حشرات تغذیه‌می‌کنند، دارای توانایی‌های شناور قوی هستند و می‌توانند به موفقیت فوق العاده‌ای در شکار دست یابند.

در الگوریتم کورکور بال سیاه BKA، ایجاد مجموعه‌ای از راه‌حل‌های تصادفی اولین گام در مقداردهی اولیه جمعیت است. ماتریس (۶)، را می‌توان برای نشان‌دادن مکان هر کورکور بال سیاه (BK) استفاده‌کرد:

$$BK = \begin{bmatrix} BK_{1,1} & BK_{1,2} & \cdots & \cdots & BK_{1,dim} \\ BK_{2,1} & BK_{2,2} & \cdots & \cdots & BK_{2,dim} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ BK_{pop,1} & BK_{pop,2} & \cdots & \cdots & BK_{pop,dim} \end{bmatrix} \quad (۶)$$

که در آن pop تعداد راه‌حل‌های بالقوه است، و $BK_{i,j}$ بعد j امین کورکور بال سیاه iام است و dim ابعاد بردارهای ویژگی است. عقاب‌های سیاه بال به عنوان شکارچی پستانداران و حشرات کوچک علفزار، بال‌ها و زوایای دم خود را با توجه به سرعت باد در طول پرواز تنظیم می‌کنند، به آرامی شناور می‌شوند تا طعمه را مشاهده‌کنند و سپس به سرعت شیرجه‌می زنند و حمله‌می‌کنند. این استراتژی شامل رفتارهای مختلف حمله برای کاوش و جستجوی سراسری است. شکل ۳، صحنه‌ای از یک کورکور بال سیاه را نشان‌می‌دهد که در هوا معلق است و بال‌های خود را بازکرده و تعادل را حفظ‌می‌کند. شکل ۴، صحنه هجوم کورکور بال سیاه به سمت طعمه خود را با سرعت بسیار زیاد نشان‌می‌دهد. رفتار جستجو و آمادگی برای حمله در الگوریتم کورکور بال سیاه در (۷)، نمایش داده شده است:



شکل ۴: رفتار حمله در الگوریتم کورکور بال سیاه [۱۵].

$$y_{t+1}^{i,j} = \begin{cases} y_t^{i,j} + n(1+\sin(r)) \times y_t^{i,j} & p > r \\ y_t^{i,j} + n \times (2r-1) \times y_t^{i,j} & else \end{cases} \quad (۷)$$

که در آن $y_t^{i,j}$ برابر بعد j ام از راه‌حل i ام در تکرار t است. r یک عدد تصادفی بین صفر و یک و p یک ثابت عددی در حدود ۰/۹ است. مقدار n را می‌توان مطابق (۸) محاسبه نمود:

$$n = 0.05 \times e^{-2 \times \left(\frac{1}{T}\right)^2} \quad (۸)$$

مهاجرت پرندگان یک رفتار پیچیده است که تحت تأثیر عوامل محیطی مانند آب و هوا و تامین غذا قرار دارد. مهاجرت پرندگان برای انطباق با تغییرات فصلی است و بسیاری از پرندگان در زمستان از شمال به جنوب مهاجرت‌می‌کنند تا شرایط و منابع بهتری برای زندگی به‌دست‌آورند. مهاجرت معمولاً توسط رهبران هدایت‌می‌شود و مهارت‌های ناوبری آنها برای موفقیت تیم بسیار مهم است. آنها فرضیه‌ای را بر اساس مهاجرت پرندگان پیشنهاد می‌کنند به گونه‌ای که اگر ارزش شایستگی جمعیت فعلی کمتر از جمعیت تصادفی باشد، رهبر، رهبری را رها می‌کند و به جمعیت مهاجر می‌پیوندد، و این نشان‌دهنده آن است که هدایت جمعیت به جلو مناسب نیست. بر عکس، اگر ارزش شایستگی جمعیت فعلی بیشتر از جمعیت تصادفی باشد، جمعیت را تا رسیدن به مقصد هدایت می‌کند. این استراتژی می‌تواند به صورت پویا رهبران عالی را برای اطمینان از مهاجرت موفق انتخاب کند. شکل ۵، تغییرات پرنده پیشرو در روند مهاجرت کورکورهای بال سیاه را نشان می‌دهد. در (۹) و (۱۰) یک مدل ریاضی برای رفتار مهاجرت کورکورهای بال سیاه ارائه شده است:

$$y_{t+1}^{i,j} = \begin{cases} y_t^{i,j} + C(0,1) \times (y_t^{i,j} - L_i^j) & F_i < P_{ri} \\ y_t^{i,j} + C(0,1) \times (L_t^j - m \times y_t^{i,j}) & \text{else} \end{cases} \quad (۹)$$

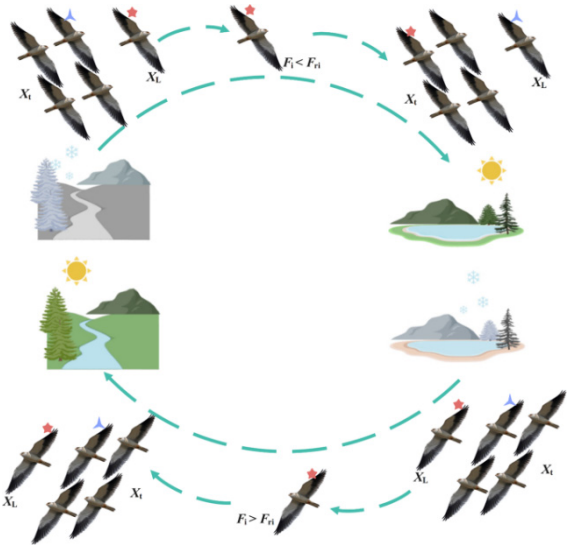
$$m = 2\sin(r + \pi/2) \quad (۱۰)$$

در این معادله، حداکثر تکرار برابر T است. برای بهبود جستجوی محلی الگوریتم کورکور بال سیاه پیشنهاد می‌شود که عقاب‌ها پیرامون نقطه ثقل جمعیت و میانگین را مانند (۱۱) و (۱۲) نیز جستجو نمایند:

$$y_{t+1}^{i,j} = y_t^{i,j} + (y^* - y_t^{i,j}) rand(0,1) \quad (۱۱)$$

$$y_{t+1}^{i,j} = y_t^{i,j} + (\bar{y} - y_t^{i,j}) rand(0,1) \quad (۱۲)$$

در معادلات اخیر y^* و $\bar{y}$ به ترتیب مقدار بهینه و متوسط و میانگین جمعیت است. برای دودویی نمودن راه‌حل‌ها می‌توان از توابع S و V استفاده نمود که دو مورد از آنها در (۱۳) و (۱۴) آمده است.



شکل ۵: رفتار جستجو و کاوش در الگوریتم کورکور بال سیاه [۱۵].

$$S(y_{l,d}(t+1)) = \frac{1}{1+\exp(-a y_{l,d}(t+1))} \quad (13)$$

$$V\left(y_{l,A}(t+1)\right) = \left|\frac{2}{\pi}\arctan\left(\frac{\pi}{2} y_{l,A}(t+1)\right)\right| \quad (14)$$

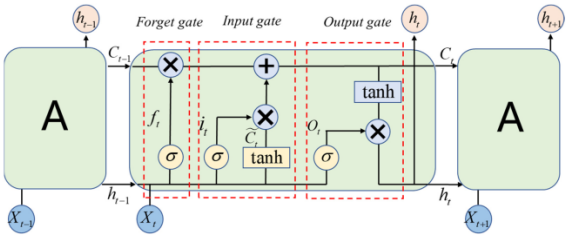
هر کدام از این دو تابع باعث می‌شود تا محدوده ویژگی‌ها بین صفر و یک نرمالیزه شود و توسط (۱۵) و (۱۶) مقادیر نرمال شده به صفر و یک نگاشت داده می‌شوند.

$$y_{l,d}(t+1) = \begin{cases} 1, & \text{if } S\left(y_{l,A}(t+1)\right) > r \\ 0, & \text{otherwise} \end{cases} \quad (15)$$

$$y_{l,d}(t+1) = \begin{cases} 1 - y_{l,d}(t), & \text{if } V\left(y_{l,d}(t+1)\right) \geq r \\ y_{l,d}(t), & \text{otherwise} \end{cases} \quad (16)$$

۳-۵ طبقه‌بندی‌کننده بهینه

در مرحله قبلی توسط الگوریتم BKA، مهمترین ویژگی‌های مجموعه داده انتخاب و استخراج‌می‌شود و به عنوان ورودی شبکه LSTM به عنوان طبقه‌بندی‌کننده بدافزار خوش‌خیم و بدخیم درنظرگرفته‌می‌شود. شبکه عصبی حافظه کوتاه‌مدت (LSTM) نوع خاصی از شبکه عصبی بازگشتی (RNN) است که شامل ساختار چرخه زمانی در یادگیری عمیق است و می‌تواند داده‌های سری زمانی را به دقت مدل‌کند. در مقایسه با RNN، مدل LSTM سه نوع دروازه برای کنترل انتقال اطلاعات بین نورون‌ها دارد که می‌تواند به طور موثر از مشکل ناپدیدشدن گرادیان RNN جلوگیری‌کند. شکل ۶ معماری اصلی LSTM نشان می‌دهد [۳۷]. همانطور که در شکل مشاهده می‌شود یک نورون LSTM منفرد شامل سه واحد کنترل گیت دروازه فراموشی، گیت ورودی و دروازه خروجی است. دروازه فراموشی اطلاعات رهاشده از نورون را تعیین‌می‌کند که منعکس‌کننده تأثیر اطلاعات تاریخی بر ارزش حالت نورون فعلی است. دروازه ورودی برای کنترل اثر داده‌های ورودی فعلی بر روی مقدار حالت نورون استفاده‌می‌شود. ابتدا اطلاعات وضعیت پنهان قبلی و ورودی فعلی را دریافت و سپس اطلاعاتی را که باید به‌روزشود، محاسبه‌می‌کند. معادلات شبکه عصبی LSTM را می‌توان به صورت (۱۷) تا (۲۲) خلاصه کرد:



شکل ۶ ساختار شبکه عصبی LSTM [۳۷].

$$f_t = \sigma(W_f\left[X_t, h_{t-1}\right]' + b_f) \quad (17)$$

$$i_t = \sigma(W_i\left[X_t, h_{t-1}\right]' + b_i) \quad (18)$$

$$C = \tanh(W_c\left[X_t, h_{t-1}\right]' + b_c) \quad (19)$$

$$C_t = f_t C_{t-1} + i_t C_t \quad (20)$$

$$o_t = \sigma(W_o\left[X_t, h_{t-1}\right] + b_o) \quad (21)$$

$$h_t = o_t \tanh(C_t) \quad (22)$$

که در این معادلات h_{t-1} خروجی نورون قبلی، X_t بردار ورودی، σ تابع فعال‌سازی، W_f، W_i و C_{t-1} و C_t به‌ترتیب وضعیت ذخیره‌سازی اطلاعات نورون قبلی و نورون فعلی است. گیت خروجی داده‌های خروجی نورون فعلی تعیین می‌کند. تابع فعال‌سازی بر اساس مقدار دروازه خروجی و وضعیت سلول فعلی محاسبه می‌شود. در معادلات W_o ضریب وزن و b_o بایاس هستند. همچنین o_t خروجی نورون فعلی و h_t حالت فعلی هستند.

یکی از چالش‌های LSTM وجود تعداد زیادی پارامتر مانند نرخ یادگیری، تعداد لایه‌های پنهان، تعداد نورون‌ها در هر لایه و غیره است. برای بهینه‌سازی پارامترهای LSTM نیاز به الگوریتمی است که پیچیدگی زیادی نداشته‌باشد زیرا پیچیدگی زیاد باعث‌می‌شود سربار محاسبات و یادگیری LSTM افزایش داده شود. الگوریتم AOA بر خلاف بسیاری از الگوریتم‌های فرابتکاری پیچیدگی زیادی ندارد و فقط چهار معادله دارد. برای بهینه‌سازی فراپارامترهای مدل LSTM نیاز است که الگوریتم فرابتکاری، جستجوی اکتشافی و بهره‌برداری (محلی) را با هم ارایه دهد. الگوریتم AOA با پارامترهای مانند MOA و MOP به‌خوبی بین جستجوی اکتشافی و محلی توازن را رعایت می‌کند. به طور کلی برای کاهش دادن خطای پیش‌بینی و طبقه‌بندی LSTM، فراپارامترهای LSTM توسط الگوریتم AOA، بهینه‌سازی می‌شود. در الگوریتم AOA از چهار عملگر ضرب، تقسیم، تفریق و جمع برای به روزرسانی معادلات استفاده شده است. در این الگوریتم پارامترهای MOA و MOP مطابق (۲۳) و (۲۴) در هر تکرار الگوریتم به‌روزرسانی می‌شوند:

$$MOA(t) = Min + \left(\frac{Max - Min}{T_{\max}}\right)t \quad (23)$$

$$MOP(t) = 1 - \left(\frac{t}{T_{\max}}\right)^{\frac{1}{\alpha}} \quad (24)$$

که در آن Max و Min به‌ترتیب بیشینه و کمینه سرعت حرکت راه‌حل‌ها به سمت جواب بهینه هستند و t شمارنده تکرار الگوریتم AOA است و Tmax حداکثر تکرار الگوریتم AOA است. مقدار ضریب MOP

بر حسب پارامتر α تعیین می‌شود که α در حدود ۵ است. از (۲۵)، برای انجام جستجوی اکتشافی با استفاده از ضرب و تقسیم و از (۲۶)، برای جستجوی محلی با استفاده از جمع و تفریق استفاده می‌شود:

$$x_{i,j}^{t+1}=\begin{cases} x^*/\left((MOP+\varepsilon)[\mu(UB_j-LB_j)+LB_j]\right) & r>0.5 \\ x^*MOP[\mu(UB_j-LB_j)+LB_j] & r\geq 0.5 \end{cases} \quad (۲۵)$$

$$x_{i,j}^{t+1}=\begin{cases} x^*-(MOP[\mu(UB_j-LB_j)+LB_j] & r>0.5 \\ bx^*MOP[\mu(UB_j-LB_j)+LB_j] & r\leq 0.5 \end{cases} \quad (۲۶)$$

که در این معادلات، x^* بهینه‌ترین پارامترهای LSTM است. UB_j و LB_j محدود بالا و پایین بعد j ام راه‌حل نظیر i ام، μ پارامتر کنترلی و یک عدد ثابت، ε یک عدد بسیار کوچک برای عدم حالت ایجاد تقسیم بر صفر است. $x_{i,j}^{t+1}$ مقدار راه‌حل i ام در بعد j ام و در تکرار جدید است و r یک عدد تصادفی در بازه صفر و یک است.

٤- نتایج تجربی

برای پیاده‌سازی روش پیشنهادی از محیط برنامه‌نویسی پایتون و کتابخانه Keras و Tensorfolw استفاده‌شده‌است. نوع اعتبارسنجی روش پیشنهادی از نوع متقاطع است و ۷۰ درصد از داده‌ها آموزشی و ۳۰ درصد داده‌ها به نسبت مساوی تقسیم‌شده و به دو دسته آزمون و اعتبارسنجی تقسیم‌می‌شوند. تعداد تکرار آزمایشات برابر ۳۰ می‌باشد و اندازه جمعیت اولیه و تعداد تکرار الگوریتم‌های فراابتکاری به ترتیب ۲۵ و۵۰ است. نرمال‌سازی مجموعه داده در بازه [۰,۱] انجام شده است و نوع تابع فعالیت نیز سیگموید است. مقدار μ برابر ۰/۵ و مقدار ε برابر ۰/۰۰۰۱ در الگوریتم AOA درنظرگرفته‌می‌شود. در این رویکرد پیشنهادی مقادیر Learning rate برابر ۰/۱، Epochs برابر ۸۰، Batch size برابر ۳۲، Dropout برابر ۰/۲۴ و تعداد نورون‌ها در لایه پنهان برابر ۲۰ است و با الگوریتم AOA بهینه انتخاب می‌شوند.

٤-١ مجموعه داده

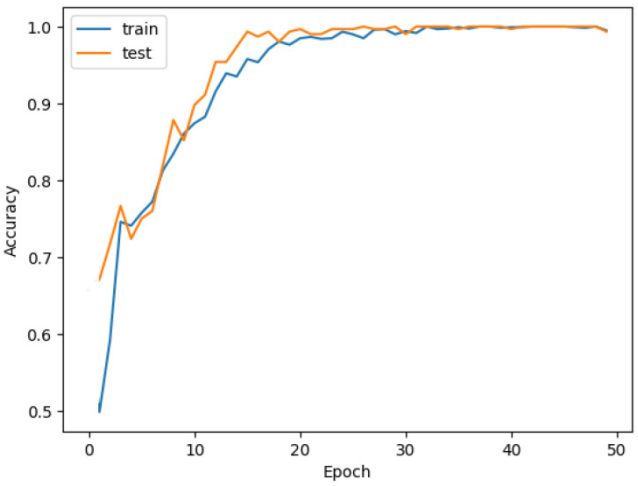
مؤسسه کانادایی امنیت سایبری یکی از جامع‌ترین و منسجم‌ترین مجموعه‌داده‌ها در تشخیص بدافزار اندروید به نام CICandMal۲۰۱۷ را ارائه می‌کند. در این مجموعه‌داده حدود۴۰۰۰ برنامه بدافزار از منابع مختلف مانند Contagiodumpst و VirusTotal جمع‌آوری شده است [۳۴].

٤-٢ معیارهای ارزیابی

برای ارزیابی روش پیشنهادی در تشخیص بدافزار اندروید از شاخص‌های دقت[1]، حساسیت[2] و صحت[3] استفاده می‌شود و در معادلات (۲۷) تا (۲۹) ارائه شده‌اند.

$$Accuracy=\frac{TP+TN}{TP+TN+FP+FN}\times ٪۱۰۰ \quad (۲۷)$$

$$Sensitivity=\frac{TP}{TP+FN}\times ٪۱۰۰ \quad (۲۸)$$



شکل ۷: همگرایی دقت به مقادیر بالا بر حسب آموزش در یک اجراء.

$$Precision=\frac{TP}{TP+FP}\times ٪۱۰۰ \quad (۲۹)$$

هر کدام از شاخص‌های ارزیابی TP، TN، FP و FN به صورت ذیل تعریف می‌گردند:

- نمونه‌های صحیح مثبت[4] (TP): بسته نرم‌افزاری از نوع بدافزار است و روش پیشنهادی، به درستی در دسته بدافزار قرار داده است.
- نمونه‌های غلط مثبت[5] (FP): بسته نرم‌افزاری از نوع غیر بدافزار است و روش پیشنهادی، به اشتباه در دسته بدافزار قرار داده است.
- نمونه‌های صحیح منفی[6] (TN): بسته نرم‌افزاری از نوع غیر بدافزار است و روش پیشنهادی، به درستی در دسته غیر بدافزار قرارداده‌است.
- نمونه‌های غلط منفی[7] (FN): بسته نرم‌افزاری از نوع بدافزار است و روش پیشنهادی، به اشتباه در دسته غیر بدافزار قرار داده است.

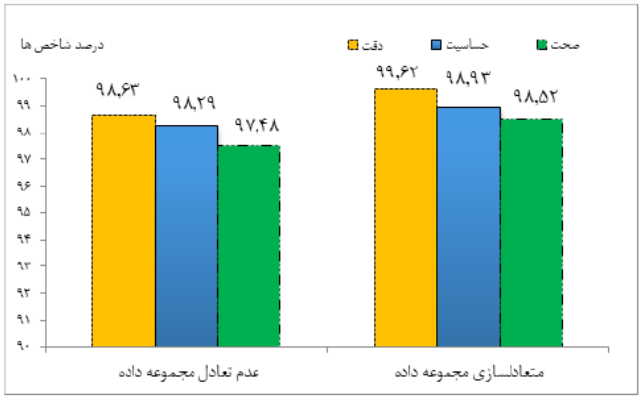
٤-٣ تحلیل و ارزیابی

در شکل ۷، نمودار همگرایی دقت در داده‌های آموزش و آزمون برحسب تکرار مراحل آموزش LSTM نشان داده شده است. ارزیابی‌ها نشان می‌دهد که دقت سمت مقدار ۱ میل می‌کند. در این آزمایش از داده‌های متعادل‌سازی‌شده با روش GAN استفاده گردیده است و به دلیل متعادل‌سازی به طور متوسط دقت، حساسیت و صحت روش پیشنهادی به ترتیب برابر ۹۹/۶۲٪، ۹۸/۹۳٪ و ۹۸/۵۲٪ است. عدم تعادل مجموعه داده نیز باعث می‌شود روش پیشنهادی به دقت، حساسیت و صحت ۹۸/۶۳٪، ۹۸/۲۹٪ و ۹۷/۴۸٪ برسد.

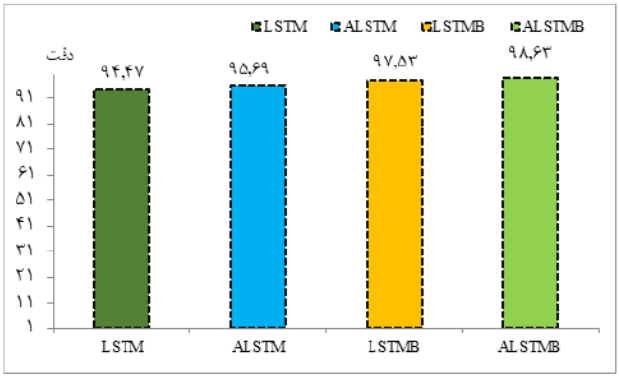
شکل ۸، تأثیر عامل متعادل‌سازی مجموعه‌داده را بر شاخص دقت، حساسیت و صحت، نشان‌می‌دهد، به طوری که اگر از متعادل‌سازی با روش GAN انجام‌شود رویکرد پیشنهادی موفق‌شده شاخص دقت، حساسیت و صحت را به ترتیب ۰/۹۹٪، ۰/۶۴٪ و ۱/۰۴٪ افزایش داده شود. به‌کارگیری GAN فقط برای افزایش دقت روش پیشنهادی نمی‌باشد بلکه این روش باعث می‌شود تا بتوان بدافزارهای پیچیده و

1. Accuracy
2. Sensitivity
3. Precision

4. True Positive
5. False Positive
6. True Negative
7. Flase Negative



شکل ۸: مقایسه حالت عدم تعادل و متعادل‌سازی مجموعه‌داده در تشخیص بدافزار.



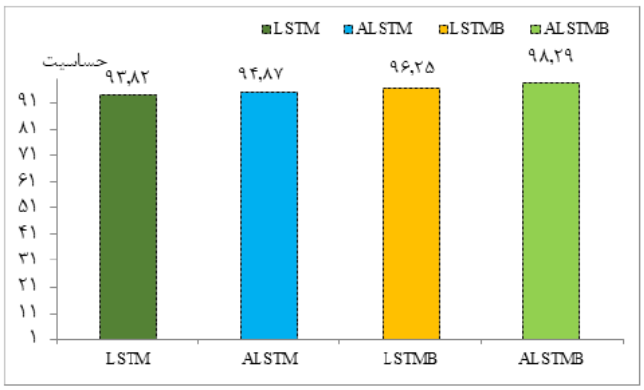
شکل ۹: مقایسه دقت حالت‌های مختلف در تشخیص بدافزار.

ناشناخته را به مجموعه‌داده اضافه نمود و سپس توانایی رویکرد پیشنهادی را برای تشخیص حملات پیچیده افزایش داد.

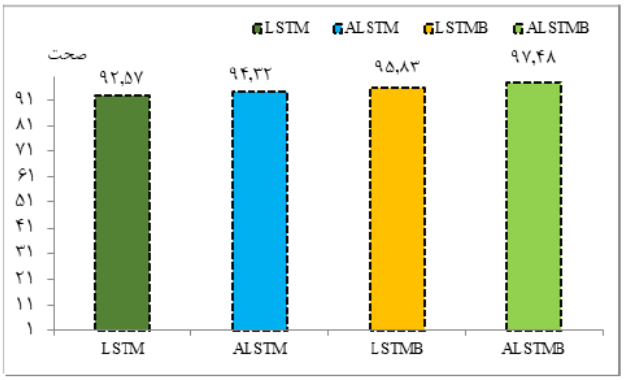
آزمایش‌ها در مجموعه‌داده اندروید در چند حالت درنظرگرفته شده است که به ترتیب در این حالتها فقط LSTM استفاده‌شده و در حالتهای دیگر الگوریتم کورکور بال سیاه و AOA استفاده می‌شود. چهار حالت ایجادشده به ترتیب LSTM، ALSTM، LSTMB، ALSTMB است. به عنوان نمونه ALSTM در این روش فقط پارامترها بهینه‌شده‌است و در LSTMB فقط انتخاب ویژگی بهینه‌شده‌است و در حالت ALSTMB همزمان پارامترها و انتخاب ویژگی هم انجام‌شده‌است. شاخص دقت، حساسیت و صحت در هر چهار حالت در شکل‌های ۹ تا ۱۱ با هم مقایسه شده‌اند.

آزمایش‌ها نشان‌می‌دهد که اگر فقط LSTM استفاده شود آنگاه دقت برابر ۹۴/۴۷٪ است و اگر از الگوریتم محاسبات ریاضی برای بهینه‌سازی پارامترهای LSTM استفاده‌شود آنگاه دقت برابر ۹۵/۶۹٪ خواهد شد و مقداری در حدود ۱/۲۲ درصد ۱/۲۲٪ افزایش خواهد یافت. اگر از الگوریتم انتخاب ویژگی کورکور بال سیاه استفاده‌شود آنگاه دقت باز افزایش‌خواهدیافت و به ۹۷/۵۳٪ خواهد رسید. به عبارت بهتر با انتخاب ویژگی دقت LSTM به حدود ۳/۰۶٪ افزایش خواهد یافت. استفاده همزمان از بهینه‌سازی پارامترهای LSTM و انتخاب ویژگی باعث افزایش بیشتر دقت به حدود ۹۸/۶۳٪ می‌شود.

بررسی شاخص‌های حساسیت و صحت نیز نشان‌می‌دهد تاثیر انتخاب ویژگی در افزایش دقت LSTM بیشتر از بهینه‌سازی پارامترهای آن است. به عبارت بهتر اگر انتخاب ویژگی انجام‌شود، تاثیر بیشتری در تشخیص بدافزار احساس‌می‌شود و اگر بیهینه‌سازی پارامترها با انتخاب ویژگی ترکیب شود آنگاه دقت LSTM از همه حالات بیشتر خواهد شد.



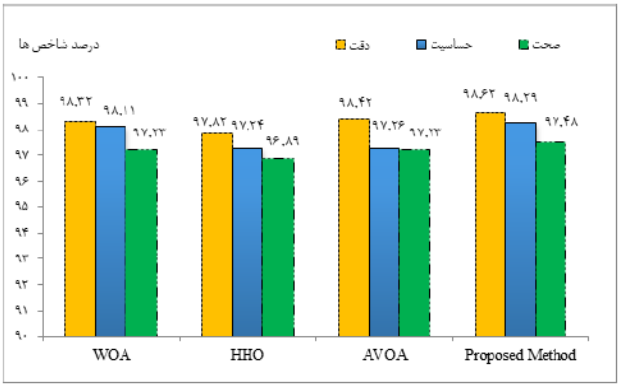
شکل ۱۰: مقایسه حساسیت روش پیشنهادی در تشخیص بدافزار.



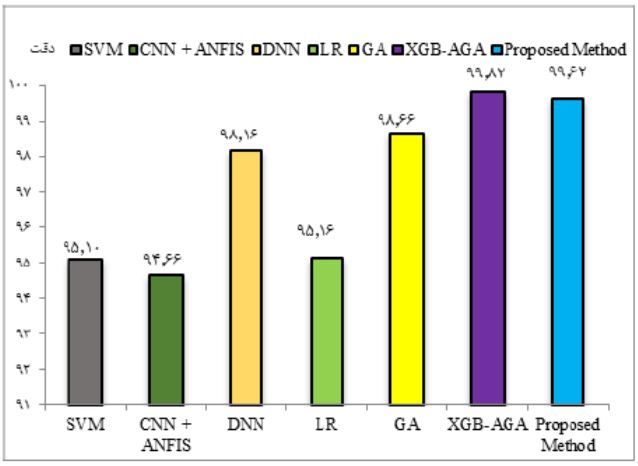
شکل ۱۱: مقایسه صحت حالت‌های مختلف در تشخیص بدافزار.

در شکل ۱۲، شاخص دقت، حساسیت و صحت روش پیشنهادی با الگوریتم بهینه‌سازی وال یا WOA، الگوریتم بهینه‌سازی شاهین یا HHO و الگوریتم بهینه‌سازی کرکس و AVOA با هم مقایسه‌شده‌است. در این آزمایش‌ها طبقه‌بندی‌کننده LSTM و پارامترهای آن توسط الگوریتم AVOA بهینه و برای انتخاب ویژگی از الگوریتم WOA، HHO و AVOA استفاده می‌شود تا توانایی آنها برای کاهش ابعاد با روش پیشنهادی مقایسه شود.

ارزیابی‌ها نشان داد که الگوریتم کورکور بال سیاه در شاخص دقت، حساسیت و صحت از روش‌های انتخاب ویژگی و کاهش ابعاد از جمله الگوریتم وال، الگوریتم شاهین و الگوریتم کرکس افریقایی عملکرد بهتری دارد که دلیل آن هوشمندی بیشتر الگوریتم در انتخاب ویژگی وکاهش ابعاد است. الگوریتم بهینه‌سازی کرکس افریقایی بعد از روش پیشنهادی در کاهش ابعاد رتبه دوم را دارد و بدترین عملکرد در کاهش ابعاد مرتبط با الگوریتم شاهین است. روش پیشنهادی با نتایج پژوهش [۳۵]، برای تشخیص بدافزار اندروید در مجموعه داده CCCS-CIC-AndMal-2020 مقایسه شده است تا کارایی آن با روش‌های مشابه مقایسه شود. در این پژوهش از روش‌های یادگیری ماشین و یادگیری عمیق از جمله SVM، CNN-ANFIS، DNN، RF، GA، Voting، استفاده شده ست. شاخص دقت روش پیشنهادی با روش‌های تشخیص بدافزار در [۳۵]، در نمودار شکل ۱۳، با هم مقایسه شده‌اند. آزمایش‌ها نشان‌می‌دهد دقت روش پیشنهادی اگر از متعادل‌سازی مجموعه‌داده با GAN استفاده نماید دارای دقتی برابر ۹۹/۶۲٪ است و اگر از متعادل‌سازی استفاده نکند دقتی برابر ۹۸/۶۳٪ دارد. دقت روش‌های SVM، CNN-ANFIS، DNN، RF، GA، XGB-AGA در تشخیص بدافزار به ترتیب دقتی برابر ۹۵/۱۰٪، ۹۴/۶۶٪ ، ۹۸/۱۶٪، ۹۵/۱۶٪، ۹۸/۱۶٪ و ۹۹/۸۲٪ دارند. آزمایش‌ها نشان می‌دهد روش پیشنهادی نسبت به هر کدام از روش‌های SVM، CNN-ANFIS، DNN، RF، GA در تشخیص بدافزار دقت



شکل ۱۲: مقایسه دقت، حساسیت و صحت در تشخیص بدافزار با روش‌های فراابتکاری در کاهش ابعاد.

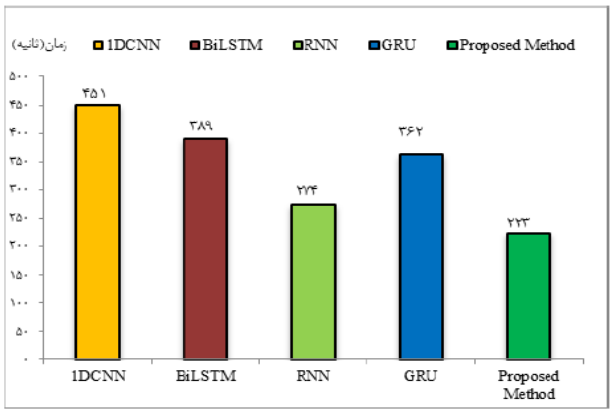


شکل ۱۳: مقایسه شاخص دقت روش پیشنهادی با روشهای یادگیری ماشین و یادگیری عمیق در تشخیص بدافزار اندروید.

بیشتری ارایه می‌دهد. روش پیشنهادی نسبت به روش XGB-AGA که دقتی برابر ۹۹٫۸۲٪ دارد به اندازه ۰٫۲٪ دقت کمتری دارد که ناچیز است اما در مقابل روش پیشنهادی به دلیل متعادل‌سازی مجموعه‌داده و اضافه‌کردن بدافزارهای ناشناخته نسبت به روش XGB-AGA توانایی بیشتری در تشخیص بدافزارهای ناشناخته‌تر دارد. در روش XGB-AGA از بهینه‌سازی ازدحام ذرات مبتنی بر تحلیل ترکیبی (PSO) و یک الگوریتم ژنتیک تطبیقی (AGA) برای تشخیص بدافزار اندروید استفاده شده است. در این روش عملکرد طبقه‌بندی‌کننده‌های یادگیری ماشین XGBoost و جنگل تصادفی (RF) با استفاده از AGA بهینه می‌شود و این موضوع نشان می‌دهد رویکرد XGB-AGA پیچیدگی بیشتری نسبت به روش پیشنهادی دارد و زمان بیشتری نیز برای تحلیل بدافزارها نیاز دارد (ضعف دیگر روشXGB-AGA نسبت به روش پیشنهادی). روش پیشنهادی نسبت به شبکه‌های عصبی کانولوشن در حدود ۳٫۹۷٪ دقت بیشتری در تشخیص بدافزار داشته است.

رویکرد پیشنهادی را می‌توان از نظر زمان اجراء با روش‌های مشابه مورد مقایسه و ارزیابی قرار داد. در شکل ۱۴ روش پیشنهادی در تشخیص بدافزار بر اساس شاخص زمان آموزش با روش‌های یادگیری عمیق مانند 1DCNN، BiLSTM، GRU و RNN مورد مقایسه قرار گرفته شده است. برای مقایسه‌ها از محیط Google Colab برای پیاده‌سازی روش-های یادگیری عمیق استفاده شده است.

ارزیابی‌ها نشان‌می‌دهد زمان آموزش روش پیشنهادی برابر ۲۲۳ ثانیه است و زمان آموزش 1DCNN، BiLSTM، GRU و RNN به ترتیب برابر ۴۵۱، ۳۸۹، ۲۷۴ و ۳۶۲ ثانیه است. به عبارت دیگر روش پیشنهادی



شکل ۱۴: مقایسه دقت، حساسیت و صحت در تشخیص بدافزار با روش‌های فراابتکاری در کاهش ابعاد.

به دلیل کاهش ابعاد ورودی LSTM با الگوریتم BKA موفق شده زمان آموزش خود را نسبت به روشهای 1DCNN، BiLSTM، GRU و RNN بیشتر کاهش دهد. ارزیابی‌ها نشان‌داد روش RNN در جایگاه دوم از نظر زمان کمینه آموزشی قراردارد و شبکه عصبی کانولوشن زمان بیشتری برای آموزش نیاز دارد.

۵- نتیجه گیری

گوشی‌های هوشمند، در حقیقت، رایانه‌های رومیزی شخصی دنیای امروز هستند. با گوشی‌های هوشمند، تقریباً می‌توانیم هر کاری را که قصد انجام آن را داریم با رایانه‌های رومیزی شخصی انجام دهیم. گوشی‌های هوشمند جزء جدایی‌ناپذیر جامعه مدرن شده‌اند و بر جنبه‌های مختلف زندگی ما تأثیر می‌گذارند. تطبیق‌پذیری و عملکرد آنها انقلابی در نحوه برقراری ارتباط، کار، سرگرمی و دسترسی به اطلاعات ما ایجاد کرده است. گوشی‌های هوشمند علاوه بر اینکه وسیله ارتباطی برای برقراری تماس و ارسال پیامک هستند، برای مرور اینترنت، رسانه‌های اجتماعی، ایمیل، عکاسی و فیلمبرداری، ناوبری، خرید آنلاین، بانکداری، سلامت و تناسب اندام، آموزش و یادگیری، سازمان شخصی و هوشمند استفاده می‌شود. در میان انواع گوشی‌های هوشمند موجود در بازار، گوشی‌های هوشمند با سیستم عامل اندروید از محبوب‌ترین‌ها هستند. دلیل محبوبیت این است که اندروید یک پلتفرم منبع باز است که بسیاری از تولیدکنندگان آن را اتخاذ می‌کنند. با وجود کاربردی بودن سیستم عامل اندروید، این سیستم عامل می‌تواند آماج حملاتی مانند بدافزار قرار بگیرد و امنیت کاربران را مختل نماید. تجزیه و تحلیل بدافزار تکنیکی است که به وسیله آن می‌توان عملکرد و منبع بدافزار، تجزیه و تحلیل و بدافزار تشخیص داده شود. در این مقاله برای تشخیص بدافزار در اندروید از یادگیری عمیق مبتنی بر LSTM استفاده می‌شود. برای افزایش اثر بخشی LSTM و کاهش خطای تشخیص بدافزار اندروید، ورودی‌های LSTM با الگوریتم کورکور بال سیاه کاهش ابعاد داده می‌شود و سپس پارامترهای LSTM نیز با الگوریتم AOA بهینه‌سازی‌می‌شود. آزمایش‌ها نشان‌می‌دهد روش پیشنهادی از روش‌های یادگیری عمیق مانند CNN، LSTM و DNN در تشخیص بدافزار اندروید دقیق‌تر است. از پیشنهادات آتی کاهش ابعاد با استفاده از شبکه عصبی CNN و طبقه‌بندی با LSTM است. هم‌زمان نیز می‌توان برای بهبود دقت پارامترهای مدل CNN-LSTM از نسخه‌های بهبودیافته الگوریتم BKA استفاده نمود. بکارگیری شبکه BERT برای استخراج ویژگی و تلفیق آن با مدل‌های یادگیری عمیق CNN و LSTM از پیشنهادهای کارهای آینده است.

مراجع

[1] X. Xiang, *et al.*, "AppChainer: Investigating the chainability among payloads in android applications," *Cybersecurity*, vol. 6, no. 16, 2023.

[2] Statista, *Number of Apps Available in Leading App Stores as of 4th Quarter 2020, 2021*, https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores, [Accessed 2022].

[3] K. A. Kumar, A. Raman, C. Gupta, and R. R. Pillai, "The recent trends in malware evolution, detection and analysis for Android devices," *Journal of Engineering Science & Technology Review*, vol. 13, no. 4, pp. 240-248, 2020.

[4] BBC, *One Billion Android Devices at Risk of Hacking, 2021*, https://www.bbc.com/news/technology-51751950, [Accessed 2022].

[5] Z. Muhammad, F. Amjad, Z. Iqbal, A. R. Javed, & T. R. Gadekallu, "Circumventing Google Play vetting policies: A stealthy cyberattack that uses incremental updates to breach privacy" *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 5, pp. 4785-4794, 2023.

[6] O. A. Alzubi, J. A. Alzubi, A. M. Al-Zoubi, M. A. Hassonah, and U. Kose, "An efficient malware detection approach with feature weighting based on Harris Hawks optimization," *Cluster Computing*, vol. 25, no. 1, pp. 1-19, Aug. 2022.

[7] H. H. R. Manzil and S. Manohar Naik, "Android malware category detection using a novel feature vector-based machine learning model," *Cybersecurity*, vol. 6, no. 1, 2023.

[8] H. AlOmari, Q. M. Yaseen, and M. A. Al-Betar, "A comparative analysis of machine learning algorithms for android malware detection," *Procedia Computer Science*, vol. 220, pp. 763-768, 2023.

[9] S. K. Smmarwar, G. P. Gupta, and S. Kumar, "Android malware detection and identification frameworks by leveraging the machine and deep learning techniques: A comprehensive review," *Telematics and Informatics Reports*, vol. 14, Article ID: 100130, Jun. 2024.

[10] S. Aurangzeb, and M. Aleem, "Evaluation and classification of obfuscated Android malware through deep learning using ensemble voting mechanism," *Scientific Reports*, vol. 13, Article ID: 3093, 2023.

[11] Y. Wu, *et al.*, "DroidRL: Feature selection for android malware detection with reinforcement learning," *Computers & Security*, vol. 128, Article ID: 103126, May 2023.

[12] N. Xie, Z. Qin, and X. Di, "GA-StackingMD: Android malware detection method based on genetic algorithm optimized stacking," *Applied Sciences*, vol. 13, no. 4, Article ID: 2629, Feb.-2 2023.

[13] P. Kumar, and S. Singh, "Security testing of Android apps using malware analysis and XGboost optimized by adaptive particle swarm optimization," *SN Computer Science*, vol. 5, no. 1, 92, Jan. 2024.

[14] A. Djenna, A. Bouridane, S. Rubab, and I. M. Marou, "Artificial intelligence-based malware detection, analysis, and mitigation," *Symmetry*, vol. 15, no. 3, Article ID: 677, Mar. 2023.

[15] J. Wang, W. C. Wang, X. X. Hu, L. Qiu, and H. F. Zang, "Black-winged kite algorithm: a nature-inspired meta-heuristic for solving benchmark functions and engineering problems," *Artificial Intelligence Review*, vol. 57, Article ID: 58, 53 pp., 2024.

[16] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, and A. H. Gandomi, "The arithmetic optimization algorithm," *Computer Methods in Applied Mechanics and Engineering*, vol. 376, Article ID: 113609, Apr. 2021.

[17] S. Nethala, *et al.*, "A Deep Learning-Based ensemble framework for robust Android malware detection," *IEEE Access*, vol. 13, pp. 46673-46696, 2025.

[18] M. U. Rashid, *et al.*, "Hybrid Android malware detection and classification using deep neural networks," *International Journal of Computational Intelligence Systems*, vol. 18, no. 1, pp. 1-26, 2025.

[19] M. S. Wasif, M. P. Miah, M. S. Hossain, M. J. Alenazi, and M. Atiquzzaman, "CNN-ViT synergy: An efficient Android malware detection approach through deep learning," *Computers and Electrical Engineering*, *pt. A,* vol. 123, Article ID: 110039, Apr. 2025.

[20] H. Kauser. Sk, & M. Anu. V, "Hybrid deep learning model for accurate and efficient android malware detection using DBN-GRU," *PloS One*, vol. 20, no. 5, Article ID: 0310230, 2025.

[21] M. Vu Minh, H. T. Nguyen, H. V. Le, T. D. Nguyen, and X. C. Do, "A static method for detecting android malware based on directed API call," *International Journal of Web Information Systems*, vol. 21, no. 3, pp. 183-204, May 2025.

[22] Y. Zhao, S. Sun, X. Huang, & J. Zhang, "An android malware detection method using frequent graph convolutional neural networks," *Electronics*, vol. 14, no. 6, Article ID: 1151, Mar-2 2025.

[23] X. Zhang, J. Wang, J. Xu, and C. Gu, "Detection of android malware based on deep forest and feature enhancement," *IEEE Access*, vol. 11, pp. 29344-29359, 2023.

[24] E. Mbunge, B. Muchemwa, J. Batani, and N. Mbuyisa, "A review of deep learning models to detect malware in Android applications," *Cyber Security and Applications*, vol. 1, Article ID: 100014, Dec. 2023.

[25] A. S. de Oliveira, and R. J. Sassi, Chimera: An Android Malware Detection Method Based on Multimodal Deep Learning and Hybrid Analysis, TechRxiv. preprint techrxiv.13359767.v1, 2020.

[26] S. Chen, B. Lang, H. Liu, Y. Chen, and Y. Song, "Android malware detection method based on graph attention networks and deep fusion of multimodal features," *Expert Systems with Applications, pt. C*, vol. 237, Article ID: 121617, Mar. 2024.

[27] G. Aldehim, *et al.*, "Gauss-mapping black widow optimization with deep extreme learning machine for android malware classification model," *IEEE Access*, vol. 11, pp. 87062- 87070, 2023.

[28] Y. Wu, J. Shi, P. Wang, D. Zeng, and C. Sun, "DeepCatra: Learning flow-and graph-based behaviours for Android malware detection," *IET Information Security*, vol. 17, no. 1, pp. 118-130, Jan. 2023.

[29] V. Ravi, and R. Chaganti, "EfficientNet deep learning meta-classifier approach for image-based android malware detection," *Multimedia Tools and Applications*, vol. 82, no. 16, pp. 24891-24917, Jul. 2023.

[30] A. R. Nasser, A. M. Hasan, and A. J. Humaidi, "DL-AMDet: Deep learning-based malware detector for android," *Intelligent Systems with Applications*, vol. 21, Article ID: 200318, Mar. 2024.

[31] F. Ullah, X. Cheng, L. Mostarda, and S. Jabbar, "Android-IoT malware classification and detection approach using deep url features analysis," *Journal of Database Management*, vol. 34, no. 2, pp. 1-26, Jan. 2023.

[32] S. Sharma, P. Ahlawat, and K. Khanna, "DeepMDFC: A deep learning based android malware detection and family classification method," *Security and Privacy*, vol. 7, no. 2, Article ID: 347, Mar./Apr. 2024.

[33] F. Taher, O. AlFandi, M. Al-kfairy, H. Al Hamadi, and S. Alrabaee, "DroidDetectMW: A hybrid intelligent model for Android malware detection," *Applied Sciences*, vol. 13, no. 13, Article ID: 7720, Jul.-1 2023.

[34] L. Hammood, İ. A. Doğru, and K. Kılıç, "Machine learning-based adaptive genetic algorithm for Android malware detection in auto-driving vehicles," *Applied Sciences*, vol. 13, no. 9, Article ID: 5403, May-1 2023.

[35] T. Wisanwanichthan and M. Thammawichai, "A lightweight intrusion detection system for IoT and UAV using deep neural networks with knowledge distillation," *Computers*, vol. 14, no. 7, Article ID: 291, Jul. 2025.

[36] P. S. Moghaddam, A. Vaziri, S. S. Khatami, F. Hernando-Gallego, and D. Martín, "Generative adversarial and transformer network synergy for robust intrusion detection in IoT environments," *Future Internet*, vol. 17, no. 6, Article ID: 258, Jun. 2025.

[37] H. Chen, Y. Bu, K. Zong, L. Huang, and W. Hao, "The effect of data skewness on the LSTM-based mooring load prediction model," *Journal of Marine Science and Engineering*, vol. 10, no. 12, Article ID: 10121931, Dec. 2022.

محسن اقبالی در سال ۱۳۹۲ مدرک کارشناسی مهندسی نرم افزار کامپیوتر خود را از دانشگاه آزاد اسلامی واحد میبد و در سال ۱۳۹۵ مدرک کارشناسی ارشد مهندسی کامپیوتر خود را در گرایش نرم افزار از دانشگاه آزاد اسلامی واحد علوم و تحقیقات تهران (پردیس یزد) دریافت کرد و هم‌اکنون دانشجوی دکتری دانشکده مهندسی کامپیوتر در گرایش نرم‌افزار دانشگاه آزاد اسلامی واحد میبد می‌باشد. زمینه‌های تحقیقاتی مورد علاقه ایشان، شبکه‌های اینترنت اشیا و یادگیری عمیق است.

محمّدرضا ملاخلیلی میبدی تحصیلات خود را در مقاطع کارشناسی و کارشناسی ارشد مهندسی کامپیوتر به‌ترتیب در سال های ۱۳۸۰ و ۱۳۸۲ از دانشگاه های شهید بهشتی و صنعتی امیرکبیر تهران و دکترای مهندسی کامپیوتر را در سال ۱۳۹۳ در واحد علوم و تحقیقات دانشگاه آزاد به پایان رسانده است. وی هم‌اکنون عضو هیأت علمی دانشگاه آزاد اسلامی واحد میبد است. زمینه های تحقیقاتی مورد علاقه ایشان عبارتند از: شبکه‌های کامپیوتری و وب، شبکه‌های مبتنی بر نرم‌افزار، محاسبات نرم و کاربردهای آن، یادگیری والگوریتم‌ها، گراف‌های تصادفی و شبکه‌های پیچیده.

کمال میرزائی در سال ۱۳۸۰ مدرک کارشناسی خود را در رشته مهندسی کامپیوتر (گرایش سخت‌افزار) از دانشگاه علم و صنعت ایران اخذ و سپس در سال ۱۳۸۳ کارشناسی ارشد خود را در رشته مهندسی کامپیوتر (گرایش هوش‌مصنوعی) دانشگاه اصفهان به پایان رسانده و مدرک دکتری خود را در سال ۱۳۹۰ از دانشگاه علوم و تحقیقات تهران در رشته مهندسی کامپیوتر (سیستم‌های نرم‌افزاری) دریافت‌کرده‌است. وی هم‌اکنون عضو هیات علمی دانشگاه آزاد اسلامی واحد میبد بوده و زمینه فعالیت ایشان علوم شناختی، شناسایی الگو و پردازش تصویر، داده‌کاوی، شبکه‌های پیچیده، یادگیری بازنمایی، پردازش موازی، الگوریتم های تکاملی و هوش جمعی است.

.