بهبود عملکرد سیستم‌های تشخیص حملات فیشینگ مبتنی بر هم‌افزایی شبکه عصبی و الگوریتم علی بابا و چهل دزد

یونس مباشری

کارشناسی ارشد، گروه مهندسی کامپیوتر، واحد یادگار امام خمینی (ره) شهر ری، دانشگاه آزاد اسلامی، شهر ری، ایران.

رضا عصاره

استادیار، گروه مهندسی کامپیوتر، واحد یادگار امام خمینی (ره) شهر ری، دانشگاه آزاد اسلامی، شهر ری، ایران.

چکیده

یکی از حملات سایبری، حملات فیشینگ است که در سال‌های اخیر به‌سرعت افزایش یافته‌اند. مهاجمان فیشینگ، کاربران را با وب‌سایت‌های تقلبی فریب می‌دهند و کاربران را به ارائه اطلاعات محرمانه در یک وب‌سایت فیشینگ ترغیب می‌کنند. تعریف روش‌های قوی، کارآمد و به‌روز برای اکتشاف فیشینگ ضروری است. استفاده از یادگیری ماشین برای آموزش سیستمی که پیام‌های فیشینگ را تشخیص می‌دهد، برای افزایش سطح امنیت در برابر حملات سایبری ضروری است. با استفاده از یافتن وزن و بایاس‌های شبکه عصبی از طریق الگوریتم علی بابا و چهل دزد می‌توان صفحات فیشینگ را با دقت بالایی شناسایی کرد. در روش پیشنهادی برای دسته‌بندی و تشخیص حملات فیشینگ از شبکه عصبی پرسپترون چند لایه استفاده می‌شود. وزن‌های شبکه عصبی پرسپترون چند لایه از طریق الگوریتم علی بابا و چهل دزد پیدا می‌شوند. نکته مهم، انتخاب روشی است که تابع هزینه با آن محاسبه شود که شامل 'MSE'، 'RMSE' و 'Accuracy' هستند. شبیه‌سازی روش پیشنهادی از طریق نرم‌افزار متلب انجام شده است. در مجموعه داده، ویژگی‌های مختلف مربوط به وب‌سایت‌های قانونی و فیشی را شناسایی کرده و ۱۳۵۳ وب‌سایت مختلف از منابع مختلف جمع‌آوری شده است. نتایج روش پیشنهادی با طرح پایه از نظر دقت، صحت، F1_Score و منحنی AUC-ROC مقایسه می‌شوند. مطابق با نتایج به دست آمده، دقت روش پیشنهادی نسبت به روش LR به میزان ۴,۹۱ درصد، نسبت به روش ماشین بردار پشتیبان به میزان ۵,۷ درصد، نسبت به روش K نزدیک‌ترین همسایه به میزان ۳,۷۲ درصد، نسبت به روش AdaBoost به میزان ۹,۰۳ درصد، نسبت به روش پرسپترون چند لایه به میزان ۳,۵۳ درصد، نسبت به روش J48 به میزان ۲,۴۶ درصد و نسبت به روش جنگل تصادفی به میزان ۰,۷۴ درصد بهبود داشته است. همچنین روش پیشنهادی نسبت به روش‌های ترکیبی الگوریتم‌های فراابتکاری و شبکه عصبی نیز بهبود داشته است. دقت روش پیشنهادی نسبت به روش ANN − EPO به میزان ۱,۳ درصد و همچنین نسبت به روش ANN − SSA به میزان ۱,۴۱ درصد بهبود داشته است.

واژگان کلیدی: حملات سایبری، حملات فیشینگ، یادگیری ماشین، شبکه عصبی پرسپترون چند لایه، الگوریتم علی بابا و چهل دزد

۱- مقدمه

یکی از چالش‌های بزرگ در فضای سایبری وجود وب‌سایت‌های جعلی است که اطلاعات کاربران را به سرقت می‌برند (Sabahno و Safara ، ۲۰۲۲). یکی از حملات سایبری، حملات فیشینگ است (Minocha و Singh ، ۲۰۲۴)؛ که در سال‌های اخیر به‌سرعت افزایش یافته‌اند (Das و همکاران، ۲۰۲۲). اصطلاح "فیشینگ" از قیاس عملیات "phishing" گرفته شده است. عبارت "ph" از "تلفن phreaking" می‌آید که روش بسیار رایجی بود که در دهه ۱۹۷۰ برای حمله به سیستم‌های تلفن استفاده می‌شد. شبکه آنلاین آمریکا (AOL[۲]) اولین قربانی این حمله بود. علاوه بر این، فیشرها خود را به جعل هویت وب‌سایت AOL محدود نکردند، بلکه فعالانه تعداد قابل‌توجهی از دروازه‌های پرداخت، شبکه‌های اجتماعی و وب‌سایت‌های مالی را تقلید کردند (Jain و Gupta ، ۲۰۲۲). مهاجمان فیشینگ، کاربران را با وب‌سایت‌های تقلبی فریب می‌دهند و کاربران را به ارائه اطلاعات محرمانه در یک وب‌سایت فیشینگ ترغیب می‌کنند (Minocha و Singh ، ۲۰۲۲). فیشینگ بیشتر مورد ترس کاربرانی است که از خدمات تراکنشی اینترنتی استفاده می‌کنند، اگرچه مطالعات زیادی بر روی تشخیص حملات فیشینگ متمرکز شده است که دقت بالایی را نشان می‌دهد، اما این حملات در اثربخشی موردنیاز برای جلوگیری از افتادن افراد در معرض این حملات مسئله دارند (Barreiro Herrera و همکاران، ۲۰۲۲). فیشینگ برای سرقت داده‌های حساس از افراد مانند نام‌های کاربری، رمز عبور، داده‌های شخصی، جزئیات حساب بانکی، اعتبارنامه‌های مهم ورود به سیستم یا اطلاعات کارت اعتباری استفاده می‌شود (Jafar و همکاران، ۲۰۲۲). تعریف روش‌های قوی، کارآمد و به‌روز برای اکتشاف فیشینگ ضروری است (Almseidin و همکاران، ۲۰۲۲). برای اینکه بتوان حملات بالقوه را شناسایی و به اندازه کافی از کاربران محافظت کرد، لازم است اصول اساسی استراتژی‌های حمله را درک نمود(Kovač و همکاران، ۲۰۲۲)؛ بنابراین دقت سیستم‌ها به‌شدت به دانش قبلی از ویژگی‌ها بستگی دارد (Zhu و همکاران، ۲۰۲۲). تجزیه‌وتحلیل راهبردهای ضد فیشینگ را می‌توان به چند رویکرد تقسیم کرد که شامل فهرست انکار، قوانین اکتشافی و فازی هستند. در هر پاسخ، ویژگی‌ها و چالش‌های مختلفی وجود دارد. در رویکرد فهرست انکار، فهرستی از URL[۳]های مشکوک یا مخرب حفظ می‌شود و با روش‌های متمایزی مانند رأی‌دهی کاربران گردآوری شده است؛ بنابراین، مرورگر، لیست انکار را به‌عنوان یک صفحه وب باز شده جست‌وجو می‌کند تا در صورت شناسایی صفحه وب به کاربر هشدار دهد. در نهایت، کاربران باید یک لیست رد را در یک ماشین یا یک سرور ذخیره کنند (Altaher ، ۲۰۱۷). از طرف دیگر، استفاده از یادگیری ماشین برای آموزش سیستمی که پیام‌های فیشینگ را تشخیص می‌دهد، به منظور افزایش سطح امنیت در برابر حملات سایبری ضروری است(Kovač و همکاران، ۲۰۲۲). روش‌های تشخیص قبلی هنوز دارای شکاف‌های مشترکی هستند که به شرح زیر خلاصه می‌شوند (Almseidin و همکاران، ۲۰۲۲):

- برخی از روش‌های تشخیص قبلی، مقادیر زیادی از هشدارهای نادرست را ثبت می‌کردند که می‌تواند منجر به مصرف زمان و منابع شود.
- روش‌های تشخیص مانند روش‌های فازی در کارهای قبل به پیش‌پردازش زیاد داده‌ها برای به دست آوردن قوانین ضد فیشینگ نیاز داشته‌اند و همچنین نمی‌توانستند مسائل مرتبط با کمبودهای بازنمایی مبتنی بر دانش را مدیریت کنند.
- برخی از روش‌های پیشین از روش‌های فراابتکاری قدیمی و کند استفاده کرده‌اند.
- برای شناسایی این نوع حملات، روش‌های مختلف مبتنی بر یادگیری ماشین مانند ماشین بردار پشتیبان توسعه یافته‌اند؛ اما چنین روش‌هایی در هنگام استفاده از داده‌های بیشتر نمی‌توانند به دقت تشخیص بالایی دست یابند و همچنین به دلیل استفاده از متغیرهای یادگیری بیشتر، زمان آموزش آن‌ها بالا بوده است.

---

[۱] Phishing
[۲] America Online Network (Aol)
[۳] Uniform Resource Locator

در روش پیشنهادی، به منظور بهبود عملکرد سیستم‌های تشخیص حملات فیشینگ، از هم‌افزایی شبکه عصبی مصنوعی و الگوریتم علی بابا و چهل دزد استفاده می‌شود. روش شبکه عصبی موردنظر، شبکه عصبی پرسپترون چند لایه است. شبکه عصبی شامل یک سری لایه ورودی، تعدادی لایه پنهان و یک لایه خروجی است. از شبکه‌های عصبی می‌توان برای انجام محاسبات پیچیده استفاده کرد. در روش پیشنهادی، از الگوریتم علی بابا و چهل دزد برای بهینه کردن یک شبکه عصبی از طریق بهینه کردن وزن‌ها یا بایاس‌ها استفاده می‌شود؛ زیرا یک مسئله مهم در شبکه عصبی، پیدا کردن وزن‌ها یا بایاس‌ها است. هدف این است که از این طریق، دقت دسته‌بند شبکه عصبی برای تشخیص حملات فیشینگ افزایش یابد. برای این منظور نیاز به یک تابع هدف یا تابع هزینه است که از طریق حل آن توسط الگوریتم علی بابا و چهل دزد، پارامترهای وزن‌ها و بایاس‌های بهینه پیدا شوند. در روش پیشنهادی، از الگوریتم علی بابا و چهل دزد در مرجع (Braik و همکاران، ۲۰۲۲) که در سال ۲۰۲۲ منتشر شده است، استفاده می‌شود.

در ادامه، ساختار مقاله به این شرح است که در بخش دوم، پیشینه تحقیق بیان و در بخش سوم، روش پیشنهادی این مقاله معرفی می‌شود. در بخش چهارم، نتایج شبیه‌سازی مربوط به روش پیشنهادی تجزیه‌وتحلیل می‌شوند و در بخش پنجم، نتیجه‌گیری از این تحقیق ارائه خواهد شد.

۲- پیشینه تحقیق

Uplenchwar و همکاران (۲۰۲۲)، یک سیستم تشخیص حملات فیشینگ برای پیام‌های متنی (PADSTM[۱]) را ارائه کرده‌اند که بر تشخیص حملات فیشینگ در پیام‌های متنی با استفاده از یادگیری ماشین متمرکز است. از روش‌های یادگیری ماشین استفاده می‌کند که شامل دسته‌بند نایوبیز، ماشین بردار پشتیبان، جنگل تصادفی و الگوریتم K نزدیک‌ترین همسایه برای شناسایی پیام‌های فیش شده است. مزیت روش ارائه‌شده این است که عملکرد دسته‌بند جنگل تصادفی از نظر دقت و F1-score در تشخیص پیام‌های فیش شده نسبت به سایر روش‌های یادگیری ماشین برتری دارد.

Palša و همکاران (۲۰۲۲)، بر آموزش مدل‌های یادگیری ماشین با استفاده از الگوریتم‌های XGBoost و درخت تصادفی بر روی دو مجموعه داده به‌دست‌آمده با استفاده از تجزیه‌وتحلیل استاتیک و پویا نمونه‌های مخرب و خوش‌خیم واقعی تمرکز داشته‌اند. سپس میزان موفقیت آن‌ها را مقایسه نموده‌اند (هم به‌صورت متقابل و هم با الگوریتم‌های دیگر، مانند جنگل تصادفی، درخت تصمیم، ماشین بردار پشتیبان و الگوریتم‌های نایوبیز). مزیت روش ارائه شده، تعیین بهترین مدل‌های یادگیری ماشین و استفاده در برنامه MLMD است. عیب روش ارائه شده، عدم توسعه مدیریت داده‌های بزرگ و شبکه‌های عصبی کارآمد و سیستم‌های مبتنی بر مدل یادگیری عمیق برای تشخیص یک حمله فیشینگ از یک مجموعه داده ثبت شده است.

Bhagwat و همکاران (۲۰۲۲)، از روش انتخاب ویژگی فراابتکاری با استفاده از الگوریتم ژنتیک (GA[۲])، الگوریتم جست‌وجوی گرانشی (GSA[۳]) و همبستگی استفاده شده است که به‌عنوان الگوریتم CGGSA[۴] نام‌گذاری شده است. ویژگی‌های بهینه‌شده توسط دسته‌بند تقویت تطبیقی و تقویت گرادیان برای شناسایی بدافزار استفاده شده‌اند. تحلیل عملکرد چارچوب ارائه شده با استفاده از مجموعه داده‌های CICMalDroid-2020 از نظر صحت، دقت، فراخوانی و امتیاز f1 ارزیابی شده است. چارچوب ارائه شده ۹۵,۳ درصد دقت را به دست آورده است. مزیت روش ارائه شده، بهبود معیارهای صحت، دقت، فراخوانی و امتیاز f1 است. عیب روش ارائه شده، افزایش سربار به دلیل استفاده از دو روش فراابتکاری مبتنی بر تکرار است.

Alzubi و همکاران (۲۰۲۲)، یک رویکرد یادگیری ماشین جدید را برای شناسایی بدافزار معرفی و آزمایش کرده‌اند. رویکرد ارائه شده از دسته‌بند ماشین بردار پشتیبان و الگوریتم بهینه‌سازی شاهین هریس تشکیل شده است. به‌طور خاص، نقش الگوریتم

---

[۱] Phishing Attack Detection System For Text Messages
[۲] Genetic Algorithm
[۳] Gravitational Search Algorithm
[۴] Correlated Genetic Gravitational Search Algorithm

بهینه‌سازی شاهین هریس بهینه‌سازی فراپارامترهای دسته‌بند ماشین بردار پشتیبان است درحالی‌که ماشین بردار پشتیبان دسته‌بندی بدافزار را بر اساس بهترین مدل انتخاب‌شده و همچنین تولید راه‌حل بهینه برای وزن‌دهی ویژگی‌ها انجام می‌دهد.

Al-Andoli و همکاران (۲۰۲۲)، یک چارچوب جدید مبتنی بر بهینه‌سازی ازدحام ذرات را برای شناسایی بدافزار توسعه داده‌اند. در این راستا، یک روش بهینه‌سازی ترکیبی یادگیری عمیق را با بهره‌برداری از ترکیب الگوریتم‌های BP[۱] و بهینه‌سازی ازدحام ذرات برای ارائه راه‌حل‌های بهینه برای تشخیص بدافزار معرفی کرده‌اند. مزیت روش ارائه شده، بهبود اثربخشی، کارایی و مقیاس‌پذیری است. عیب روش ارائه شده، افزایش بار محاسبات با افزایش تعداد تکرار است.

Dhiyanesh و همکاران (۲۰۲۱)، یک روش انتخاب ویژگی و دسته‌بندی مؤثر برای تشخیص حملات فیشینگ در شبکه‌های بی‌سیم پیشنهاد کرده‌اند. در ابتدا، داده‌های ایمیل جمع‌آوری شده‌اند که شامل ویژگی‌های بیشتری است که باید استخراج شوند. سپس، الگوریتم EHO[۲] برای انتخاب مرتبط‌ترین ویژگی‌ها از میان تمام ویژگی‌های استخراج‌شده مربوط به ایمیل اعمال شده است.

Bhagwat و Gupta (۲۰۲۱)، یک ربات فیشینگ توییتر را با استفاده از یادگیری ماشین ساخته‌اند. آزمایشی را روی تشخیص آدرس اینترنتی فیشینگ، ایمیل‌های فیشینگ و وب‌سایت‌های فیشینگ انجام داده‌اند. برای تشخیص URL فیشینگ از دسته‌بندهای مختلفی استفاده کرده‌اند و با دقت بالاتر روی زمان‌بندی آموزش مجموعه داده تمرکز کرده‌اند. مزیت روش ارائه شده، دقت بالاتر و زمان کمتر نسبت به روش‌های مورد مقایسه است. عیب روش ارائه شده، عدم ترکیب روش‌های کاهش ویژگی برای بهبود دقت سیستم است.

Stobbs و همکاران (۲۰۲۰)، تأثیر ویژگی‌های مختلف و روش‌های بهینه‌سازی را بر دقت تشخیص حملات فیشینگ هوشمند با استفاده از الگوریتم‌های یادگیری ماشین بررسی کرده‌اند. این کار به بهینه‌سازی انتخاب ویژگی پرداخته است. برای تنظیم فراپارامتر، TPE[۳] و الگوریتم ژنتیک مورد آزمایش قرار گرفته‌اند که بهترین گزینه وابسته به مدل بوده‌اند. برای انتخاب ویژگی، الگوریتم‌های ژنتیک، بهینه‌سازی پروانه آتش و بهینه‌سازی ازدحام ذرات استفاده شده‌اند که با بهترین عملکرد برای بهینه‌سازی ازدحام ذرات با مدل جنگل تصادفی بوده است.

Abedin و همکاران (۲۰۲۰)، عملکرد سه دسته‌بند یادگیری ماشین مانند جنگل تصادفی، رگرسیون لجستیک، K نزدیک‌ترین همسایه مقایسه شده‌اند. تقسیم مجموعه داده به دو بخش، یکی برای آموزش و دیگری برای آزمایش انجام شده است. ۸۰٪ از مجموعه داده برای آموزش و ۲۰٪ از مجموعه داده برای آزمایش استفاده شده‌اند. تقسیم را با استفاده از کتابخانه Scikit-Learn در زبان برنامه‌نویسی پایتون انجام داده‌اند.

Zhang و همکاران (۲۰۱۸)، یک مدل یادگیری عمیق ترکیبی جدید را برای تشخیص حملات فیشینگ پیشنهاد کرده‌اند. این روش، شامل دو جزء AE[۴] و شبکه عصبی کانولوشن است. AE برای بازسازی ویژگی‌هایی که رابطه همبستگی بین ویژگی‌ها را به‌طور صریح افزایش می‌دهند، اتخاذ می‌شود. شبکه‌های عصبی عمیق ترکیبی را با سه الگوریتم دسته‌بندی سنتی شامل ماشین بردار پشتیبان، درخت تصمیم و LinearSVC مقایسه کرده‌اند.

Darshan و همکاران (۲۰۱۶)، بدافزار بر روی فاخته اجرا می‌شود تا رفتار زمان اجرا آن را به دست آورد. در پایان اجرا، cuckoo sandbox تماس‌های سیستمی را گزارش می‌کند که توسط بدافزار در حین اجرا فراخوانی شده‌اند. با این حال، این گزارش با فرمت JSON است و برای استخراج تماس‌های سیستمی باید به فرمت MIST تبدیل شود.

جدول (۱): خلاصه‌ای از پژوهش‌های موردمطالعه

[۱] Backpropagation
[۲] Elephant Herding Optimization
[۳] Tree-Structured Parzen Estimator
[۴] Autoencoder

منبع بررسی شده	روش ارائه شده	شکاف‌های پژوهشی	مزیت روش ارائه شده
Uplenchwar و همکاران (۲۰۲۲)	تشخیص حمله فیشینگ در پیام‌های متنی با استفاده از یادگیری ماشین	عدم استفاده از دسته‌بندهای تجمعی برای افزایش دقت	بهبود عملکرد دسته‌بند جنگل تصادفی از نظر دقت و F1-score
Palša و همکاران (۲۰۲۲)	MLMD - یک ابزار آنتی‌ویروس شناسایی بدافزار بر اساس الگوریتم یادگیری ماشین XGBoost	عدم توسعه مدیریت داده بزرگ و شبکه عصبی کارآمد و سیستم‌های مبتنی بر مدل یادگیری عمیق برای تشخیص یک حمله فیشینگ از یک مجموعه داده ثبت شده	تعیین بهترین مدل‌های یادگیری ماشین و استفاده در برنامه MLMD
Bhagwat و همکاران (۲۰۲۲)	شناسایی بدافزار با وزن‌دهی ویژگی بر اساس هوش جمعی	افزایش سربار به دلیل استفاده از دو روش فراابتکاری مبتنی بر تکرار	بهبود معیارهای صحت، دقت، فراخوانی و امتیاز f1
Alzubi و همکاران (۲۰۲۲)	یک رویکرد یادگیری ماشین جدید برای شناسایی بدافزار مبتنی بر ماشین بردار پشتیبان و شاهین هریس	عدم استفاده از رویکردهای جدیدتر مانند الگوریتم بهینه‌سازی گله اسب	اندازه‌گیری اهمیت هر ویژگی و تجزیه‌وتحلیل روابط احتمالی بین ویژگی وزن‌دار و نوع حمله بدافزار
Al-Andoli و همکاران (۲۰۲۲)	رویکرد ترکیبی مبتنی بر BPSO و یادگیری عمیق برای انتخاب ویژگی اندروید و شناسایی بدافزار	افزایش بار محاسبات با افزایش تعداد تکرار	بهبود اثربخشی، کارایی و مقیاس‌پذیری
Dhiyanesh و همکاران (۲۰۲۱)	انتخاب ویژگی و روش دسته‌بندی مؤثر برای تشخیص حملات فیشینگ	عدم بررسی یک CNN بهینه شده همراه با مجموعه ویژگی	تشخیص بهتر در مقایسه با روش‌های قبلی
Bhagwat و Gupta (۲۰۲۱)	تشخیص بدافزار اندروید با استفاده از انتخاب ویژگی‌های ترکیبی فراابتکاری و روش‌های یادگیری گروهی	عدم ترکیب روش‌های کاهش ویژگی برای بهبود دقت سیستم	دقت بالاتر و زمان کمتر نسبت به روش‌های مورد مقایسه
Stobbs و همکاران (۲۰۲۰)	تشخیص صفحه وب فیشینگ با استفاده از یادگیری ماشین بهینه	عدم استفاده از تکرارهای بیشتر برای اجرای بهینه‌سازی و افزایش دقت	بهبود دقت
Abedin و همکاران (۲۰۲۰)	تشخیص حمله فیشینگ با استفاده از روش‌های دسته‌بندی یادگیری ماشین	عدم بهبود دقت با تغییر ویژگی‌ها	بهبود دقت
Zhang و همکاران (۲۰۱۸)	تشخیص حملات فیشینگ با شبکه‌های عصبی ترکیبی	رتبه صفحه Alexa در این کار استفاده نشده است.	قابلیت تعمیم بالا
Darshan و همکاران (۲۰۱۶)	تشخیص بدافزار بر اساس گزارش تولیدشده توسط فاخته و الگوریتم یادگیری ماشین	عدم استفاده از رویکردهای بهینه‌سازی جدیدتر	بالاترین دقت، بالاترین نرخ مثبت واقعی و کمترین نرخ مثبت کاذب

۳- روش پیشنهادی

مجموعه داده مورد استفاده از سایت https://archive.ics.uci.edu/ml/datasets/Website+Phishing است.

۳-۱- استفاده از شبکه عصبی پرسپترون چند لایه و بررسی تنظیم شبکه

در روش پیشنهادی برای دسته‌بندی و تشخیص حملات فیشینگ از شبکه عصبی پرسپترون چند لایه استفاده می‌شود که یک الگوریتم دسته‌بندی باینری یادگیری تحت نظارت است. وزن‌ها و بایاس‌ها به‌صورت تصادفی مقداردهی می‌شوند. ورودی‌ها در وزن‌ها ضرب می‌شوند، مقادیر به دست آمده با هم و سپس با بایاس جمع می‌شوند. نتیجه از تابع فعال‌ساز عبور می‌کند و خروجی نورون را تشکیل می‌دهد. با مقداردهی وزن‌ها به‌صورت تصادفی، نتیجه معمولاً نامناسب می‌شود. بنابراین نیاز است که وزن‌ها تغییر کنند. تغییر وزن‌ها باید به شکلی انجام شود که خروجی‌های نورون به خروجی‌های واقعی نزدیک باشند. به فرآیند تغییر وزن‌های نورون برای رسیدن به خروجی مطلوب، یادگیری

نورون گفته می‌شود که در روش پیشنهادی برای یافتن وزن‌های بهینه از الگوریتم علی بابا و چهل دزد استفاده می‌شود. روش پیشنهادی بر اساس پرسپترون چند لایه (MLP) است که مهم‌ترین مدل از شبکه عصبی عمیق است. MLP از سه لایه یا بیشتر ساخته شده است (لایه ورودی، یک یا چند لایه پنهان و لایه خروجی). آن‌ها حاوی آستانه، وزن و تابع انتقال برای انتقال داده‌ها به لایه خروجی هستند. اگر خطا بین داده‌های شناخته شده و داده‌های لایه خروجی به اندازه هدف نباشد، آستانه لایه‌ها و وزن‌ها از عقب به جلو تنظیم می‌شود. ورودی‌ها بر اساس وزن خروجی‌های آن است. تولید خروجی y به تابعی به نام تابع فعال‌سازی نیاز دارد که ورودی‌های تشکیل شده از x1، x2، ...، xn را با وزن‌های مربوط به w1، w2، ...، wn ضرب می‌کند. پس از آن، خروجی ها را از طریق یک تابع فعال‌سازی غیر خطی قرار می‌دهد؛ که به‌صورت $\varphi(\sum_{i=1}^{n} w_i x_i + b)$ و یا $\varphi(w^T x + b)$ نوشته می شود. جایی که w نشان‌دهنده بردار وزن، x نشان‌دهنده بردار ورودی، b نشان‌دهنده بایاس و φ نشان‌دهنده تابع فعال‌سازی است.

۳-۲- حل تابع هدف با استفاده از الگوریتم علی بابا و چهل دزد

دلیل استفاده از الگوریتم علی بابا و چهل دزد در روش پیشنهادی به سه دلیل است. دلیل اول، مدل های به روز رسانی موقعیت از الگوریتم علی بابا و چهل دزد به طور موثر به افراد جمعیت کمک می کند تا هر منطقه را در فضای جست‌وجو اکتشاف و بهره برداری کنند. دلیل دوم، جست‌وجوی تصادفی که دزدان در فضای جست‌وجو استفاده می کنند، نه تنها تنوع جمعیت را افزایش می دهد، بلکه سرعت همگرایی را نیز تضمین می کند که نشان دهنده تعادل کارآمد بین اکتشاف و بهره برداری است. دلیل سوم، تعداد پارامترها در الگوریتم علی بابا و چهل دزد کم است، اما آنها توانایی خوبی برای بهبود قدرت و عملکرد آن دارند. برتری چهارم، بار محاسباتی الگوریتم علی بابا و چهل دزد کم است.

الگوریتم علی بابا و چهل دزد با ایجاد تصادفی مجموعه ای از موقعیت ها(یعنی راه حل های بالقوه)، با در نظر گرفتن کران بالا و پایین متغیرهای مسئله، بهینه سازی را در حل یک مسئله بهینه سازی آغاز می کند. پس از آن، بهترین موقعیت، بهترین موقعیت سراسری دزدان و نقشه های هوشمندانه مرجانه مقداردهی می‌شوند. کیفیت هر راه حل ایجاد شده با استفاده از یک تابع تناسب از پیش تعریف شده ارزیابی می شود که به موجب آن مناسب بودن هر راه حل در هر تکرار به منظور شناسایی دزد با راه حل بهینه، مجدداً محاسبه می شود. برای هر بعد، موقعیت جدید دزدها به صورت تکراری در هر تکرار با استفاده از رابطه (۵)، (۱۰)، (۱۱) محاسبه می‌شوند. امکان سنجی هر موقعیت جدید مورد بررسی قرار می گیرد تا مشاهده شود که آیا از منطقه جست‌وجو خارج می‌شود یا خیر. سپس موقعیت جدید، بهترین موقعیت، بهترین موقعیت سراسری دزدان و نقشه های هوشمندانه مرجانه بر این اساس ارزیابی و به روز می‌شوند. به جز مراحل اولیه، به طور مکرر انجام می شود تا زمانی که به شرایط ارزیابی خاتمه برسد. در پایان، بهترین موقعیت دزدان به عنوان راه حل مسئله بهینه سازی امتیازدهی می شود.

مقداردهی اولیه تصادفی

الگوریتم علی بابا و چهل دزد با مقداردهی اولیه تصادفی موقعیت تعدادی از n فرد در یک فضای جستجوی d بعدی مانند شکل زیر آغاز می شود (Braik و همکاران، ۲۰۲۲):

$$x = \begin{bmatrix} x_1^1 & x_2^1 & x_3^1 & \cdots & x_d^1 \\ x_1^2 & x_2^2 & x_3^2 & \cdots & x_d^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^n & x_2^n & x_3^n & \cdots & x_d^n \end{bmatrix} \quad (۱)$$

که در آن x موقعیت همه دزدان است، d تعداد متغیرهای یک مسئله معین است و x_j^i نشان دهنده بعد jام دزد است. موقعیت اولیه جمعیت (به عنوان مثال، دزدان) را می توان همانطور که در رابطه (۲) نشان داده شده است ایجاد کرد (Braik و همکاران، ۲۰۲۲).

$$x^i = l_j + r \times (u_j - l_j) \quad (۲)$$

که در آن x^i موقعیت دزد iام است که نشان‌دهنده راه‌حل کاندید برای یک مسئله است، l_j و u_j به ترتیب به کران‌های پایین و بالا در بعد jام اشاره دارند و r یک عدد تصادفی توزیع شده یکنواخت در محدوده ۰ تا ۱ است. سطح هوش مرجانه نسبت به همه دزدان را می‌توان به صورت زیر مقداردهی کرد (Braik و همکاران، ۲۰۲۲):

$$m = \begin{bmatrix} m_1^1 & m_2^1 & \cdots & m_d^1 \\ m_1^2 & m_2^2 & \cdots & m_d^2 \\ \vdots & \vdots & \vdots & \vdots \\ m_1^n & m_2^n & \cdots & m_d^n \end{bmatrix} \quad (۳)$$

جایی که m_j^i نشان دهنده سطح هوشیار مرجانه در رابطه با دزد i ام در بعد jام است.

ارزیابی برازندگی

مسئله‌ای که برای استفاده از الگوریتم فراابتکاری وجود دارد، این است که تابع هدف برای این الگوریتم مشخص گردد که با CostFunction مشخص شده است. برای روش پیشنهادی یک تابع به نام CostNNClassification نوشته شده است. با توجه به اینکه تشخیص حملات فیشینگ، یک مسئله طبقه‌بندی است، بنابراین باید طبقه‌بندی در تابع هدف تعریف شود.

تابع هدف در حالت کلی همان تابع هزینه‌ای است که قرار است الگوریتم علی بابا و چهل دزد آن را حل کند. چیزی که الگوریتم علی بابا و چهل دزد به‌عنوان متغیر می‌دهد، W ها (پارامترهای وزن‌ها و بایاس‌ها) هستند. هدف این است که الگوریتم علی بابا و چهل دزد W ها را رفته‌رفته بهینه کند؛ بنابراین ورودی، W هایی هستند که الگوریتم علی بابا و چهل دزد مشخص می‌کند.

نکته مهم، انتخاب روشی است که تابع هزینه با آن محاسبه شود که شامل 'MSE'، 'RMSE' و 'Accuracy' هستند.

مقادیر متغیرهای تصمیم در یک تابع برازندگی تعریف شده توسط کاربر درج می شود که برای موقعیت هر دزد ارزیابی می شود. مقادیر تناسب مربوطه در یک آرایه به شکل زیر ذخیره می‌شوند (Braik و همکاران، ۲۰۲۲):

$$f = \begin{bmatrix} f_1([x_1^1 & x_2^1 & \cdots & x_d^1)] \\ f_2([x_1^2 & x_2^2 & \cdots & x_d^2)] \\ \vdots & \vdots & \vdots & \vdots \\ f_n([x_1^n & x_2^n & \cdots & x_d^n)] \end{bmatrix} \quad (۴)$$

که در آن x_d^n بعد d ام موقعیت دزد n ام است.

دزدان، عامل های جست‌وجو برای حل مسئله هستند. در واقع شامل یکسری بردار صفر و یک است که الگوریتم این بردار را پیشنهاد می‌کند. این بردار در تابع ارزیابی می‌رود تا کیفیت آن ارزیابی شود. در شبیه‌سازی الگوریتم علی بابا و چهل دزد، کیفیت راه حل برای مکان جدید هر دزد بر اساس یک تابع برازندگی تعریف شده ارزیابی می شود. پس از آن، اگر موقعیت مکانی بهتر از کیفیت راه حل فعلی باشد، به روز می شود. هر دزد در صورتی که کیفیت راه حل او کارآمدتر از راه حل جدید باشد، در مکان فعلی خود می ماند.

مدل ریاضی پیشنهادی

سه مورد اساسی ممکن است هنگام جست‌وجوی دزدان برای علی بابا رخ دهد. در هر مورد، فرض بر این است که دزدان به طور موثر در محیط اطراف جست‌وجو می کنند، در حالی که نسبتی نیز به دلیل هوش مرجانه رخ می دهد که دزدان را مجبور به جست‌وجو در مکان های تصادفی می کند. رفتار جست‌وجوی فوق را می توان به صورت ریاضی به صورت زیر مدلسازی کرد: مورد ۱ دزدان

ممکن است علی بابا را با کمک اطلاعاتی که از شخصی به دست آورده اند، ردیابی کنند. در این صورت مکان های جدید دزدان را می توان به شرح زیر به‌دست آورد (Braik و همکاران، ۲۰۲۲):

$$x_{t+1}^{i} = gbest_t + \left[Td_t\left(best_t^i - y_t^i\right)r_1 + Td_t\left(y_t^i - m_t^{a(i)}\right)r_2\right] sgn(rand - 0.5); r_3 \geq 0.5, r_4 > P_{p_t} \quad (۵)$$

جایی که x_{t+1}^{i} نشان دهنده موقعیت دزد i در تکرار $(t+1)$، y_t^i موقعیت علی بابا در رابطه با دزد i در تکرار t، $best_t^i$ نشان دهنده بهترین موقعیتی است که تاکنون توسط دزد i در تکرار t به دست آورده است، $gbest_t$ نشان دهنده بهترین موقعیت سراسری است که تا کنون توسط هر دزدی به دست آمده است. $m_t^{a(i)}$ نشان دهنده سطح هوش مرجانه است که برای استتار دزد i در تکرار t استفاده می شود. Td_t، فاصله ردیابی دزدان در تکرار t است، P_{p_t} نشان دهنده پتانسیل ادراک دزدان به علی بابا در تکرار t است. حد r_1، r_2 و r_4 اعداد تصادفی هستند که با توزیع یکنواخت بین صفر و یک تولید می‌شوند، $r_3 \geq 0.5$، صفر یا یک را می دهد و نشان دهنده درستی یا نادرستی اطلاعات است. $sgn(rand - 0.5)$، ۱ یا -1 را برای تغییر جهت فرآیند جستوجو نشان می دهد.

پارامتر a در $m_t^{a(i)}$ را می توان به صورت زیر تعریف کرد (Braik و همکاران، ۲۰۲۲):

$$a = [(n-1).rand(n,1)] \quad (۶)$$

که در آن $rand(n,1)$ بردار اعداد تصادفی تولید شده با توزیع یکنواخت در محدوده [۰،۱] را نشان می دهد. مرجانه برنامه های زیرکانه خود را به روزرسانی می کند؛ در صورتی که کیفیت راه حل جدیدی که دزدان ارائه می دهند بهتر از موقعیت قبلی آنها باشد. در این مورد، رابطه (۷) می تواند برای به روز رسانی برنامه های او استفاده شود (Braik و همکاران، ۲۰۲۲).

$$m_t^{a(i)} = \begin{cases} x_t^i & if\ f\left(x_t^i\right) \geq f\left(m_t^{a(i)}\right) \\ m_t^{a(i)} & if\ f\left(x_t^i\right) < f\left(m_t^{a(i)}\right) \end{cases} \quad (۷)$$

که در آن $f(0)$ مخفف نمره تابع برازندگی است. پارامتر فاصله ردیابی Td_t همانطور که در رابطه (۸) ارائه شده است، تعریف می‌شود (Braik و همکاران، ۲۰۲۲).

$$Td_t = \alpha_0 e^{-\alpha_1 (t/T)_1^{\alpha}} \quad (۸)$$

که در آن t و T به ترتیب بیانگر تعداد فعلی و حداکثر تعداد تکرارها هستند. $\alpha_0 (\alpha_0 = 1)$ تخمین اولیه فاصله ردیابی را در اولین تکرار نشان می دهد و α_1، یک مقدار ثابت است که برای مدیریت قابلیت های اکتشاف و بهره برداری استفاده می شود. رابطه (۸) نشان می دهد که Td_t به طور تکراری در طول دوره تکرار الگوریتم علی بابا و چهل دزد به روز می شود.

فاصله ردیابی، به شدت بر توانایی جستوجو تأثیر می گذارد، که تأثیر زیادی بر قدرت اکتشاف و بهره برداری الگوریتم علی بابا و چهل دزد دارد. مقادیر زیاد Td_t منجر به جستوجوی سراسری می شود که می تواند به سمت اکتشاف بیشتر منحرف شود و این ممکن است از راه حل های بهینه محلی جلوگیری کند. از طرف دیگر، مقادیر کوچک Td_t منجر به جستوجوی محلی می شود، جایی که این باعث افزایش توانایی بهره برداری در الگوریتم علی بابا و چهل دزد می شود تا دزدان امکان خوبی برای یافتن علی بابا داشته باشند.

به طور مشابه، پارامتر پتانسیل ادراک P_{p_t} همانطور که در رابطه (۹) ارائه شده است، تعریف می شود (Braik و همکاران، ۲۰۲۲).

$$P_{p_t} = \beta_0 log\left(\beta_1 (t/T)^{\beta_0}\right) \quad (۹)$$

که در آن $\beta_0(\beta_0 = 0.1)$ تخمین تقریبی نهایی را از احتمال اینکه دزدان در پایان فرآیند تکراری الگوریتم علی بابا و چهل دزد به هدف خود دست یابند نشان می دهد و β_1، یک مقدار ثابت است که برای مدیریت قابلیت‌های اکتشاف و بهره‌برداری استفاده می‌شود.

با افزایش تدریجی مقدار P_{p_t}، الگوریتم علی بابا و چهل دزد تمایل دارد از جست‌وجوی سراسری به جست‌وجوی محلی در امیدوار کننده ترین مناطقی که راه حل بالقوه ای در این مناطق یافت می شود، حرکت کند. به عبارت دیگر، مقادیر زیاد P_{p_t} منجر به جست‌وجوی محلی می شود که جست‌وجو را در مناسب ترین مناطق فضای جست‌وجو تشدید می کند. از طرف دیگر، مقادیر کوچک امکان جست‌وجو در مجاورت راه حل های خوب فعلی را کاهش می دهد. بنابراین، افزایش این مقدار، الگوریتم علی بابا و چهل دزد را تحریک می کند تا فضای جست‌وجو را در مقیاس سراسری کشف کند و جست‌وجو را در همه مناطق فضای جست‌وجو متنوع سازد. برای تمام مسائل، α_1 و β_1 برابر ۲،۰ هستند. ممکن است دزدان متوجه شوند که فریب خورده اند، بنابراین به طور تصادفی فضای جست‌وجوی علی بابا را کشف می کنند. در این صورت مکان های جدید دزدان را می توان به شرح زیر به‌دست آورد (Braik و همکاران، ۲۰۲۲):

$$x_{t+1}^{i} = Td_t\left[(u_j - l_j)rand + l_j\right]; r_3 \geq 0.5, \quad r_4 \leq P_{p_t} \qquad (۱۰)$$

پارامتر Td_t در رابطه (۱۰) گنجانده شده است؛ زیرا دزدها از دانش خوبی برای تشخیص مناسب ترین مناطق فضای جست‌وجو که خانه علی بابا می تواند باشد، برخوردارند. مورد بعد، به منظور بهبود ویژگی‌های اکتشاف و بهره‌برداری از الگوریتم علی بابا و چهل دزد، جست‌وجو را در موقعیت‌های دیگری غیر از مواردی که می‌توان با استفاده از رابطه (۵) به دست آورد، در نظر گرفت. در این صورت می توان مکان های جدید دزدها را به شرح زیر به‌دست آورد (Braik و همکاران، ۲۰۲۲):

$$x_{t+1}^{i} = gbest_t - \left[Td_t\left(best_t^{i} - y_t^{i}\right)r_1 + Td_t\left(y_t^{i} - m_t^{a(i)}\right)r_2\right] sgn(rand - 0.5); r_3 < 0.5 \qquad (۱۱)$$

۴- شبیه‌سازی

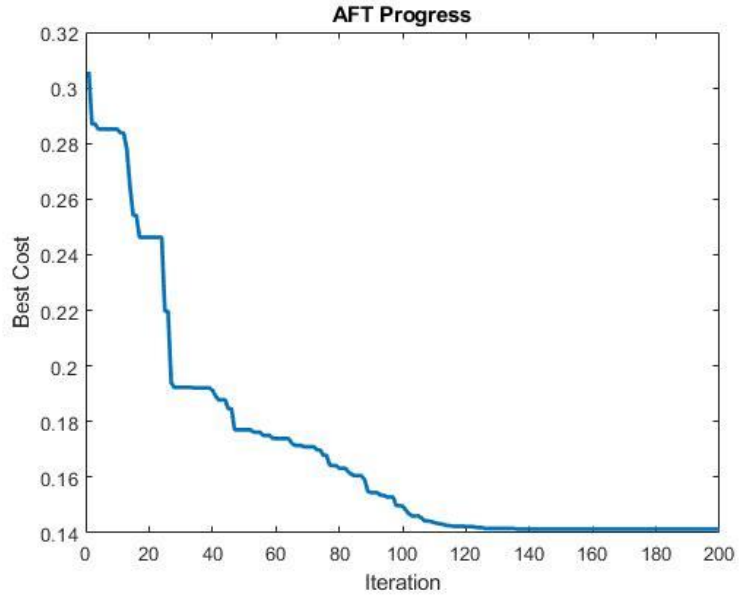
شبیه‌ساز مورد استفاده که برای شبیه‌سازی روش پیشنهادی نشان داده شده است، MATLAB است. برای تشخیص فیشینگ URL، باید عملکرد را از دیدگاه‌های مختلف مانند دقت، صحت و موارد دیگر تعیین نمود. مدل پیشنهادی بر روی ۸۰٪ از مجموعه داده‌ها آموزش داده شده و مدل بر روی ۲۰٪ از مجموعه داده‌ها آزمایش شده است. اندازه‌گیری‌های ارزیابی عملکرد، نتیجه‌ای از بهترین مدل را به دست آورد که بهترین نتایج را به دست آورده بود. زمانی که خروجی شامل دو نوع کلاس و یا بیشتر باشد، ماتریس درهم‌ریختگی ساده‌ترین راه برای اندازه‌گیری کارایی یک مسئله دسته‌بندی است. ماتریس درهم‌ریختگی یک جدول با دو بعد است (مقدار واقعی[۱] و پیش‌بینی شده[۲]). هر دو بعد دارای TP، TN، FP و FN است. TP، زمانی است که هر دو کلاس واقعی و پیش‌بینی از نقاط داده ۱ است. TN، زمانی است که هر دو کلاس واقعی و پیش‌بینی از نقاط داده صفر است. FP، زمانی است که کلاس واقعی از نقطه داده صفر و کلاس پیش‌بینی ۱ است. FN، زمانی است که کلاس واقعی از نقطه داده ۱ و کلاس پیش‌بینی صفر است. معیارهای ارزیابی عبارت اند از:

- دقت: دقت تعداد صحیح دسته‌بندی شده فیشینگ و URL های قانونی را نشان می‌دهد.
- صحت: صحت تعداد URL های فیشینگ دسته‌بندی شده به‌عنوان فیش بر تعداد کل URL های فیشینگ است.
- فراخوانی (recall): فراخوانی را می‌توان در قالب تعداد مثبت‌های بازگردانده شده توسط مدل یادگیری ماشین تعریف کرد.

[۱] Actual Value
[۲] Predicted Value

- امتیاز F1: این امتیاز، میانگین هارمونیک از دقت و فراخوانی را به دست می آورد. به بیان ریاضی، امتیاز F1_Score میانگین وزن دار از دقت و فراخوانی است.
- منحنی AUC-ROC: AUC[1]، ناحیه زیر منحنی ROC[2] است. ناحیه زیر منحنی ROC احتمالی است که در آن دسته‌بند اطمینان بیشتری کسب کند که یک نمونه مثبت که به‌طور تصادفی انتخاب شده است در واقع مثبت‌تر از آن است که یک نمونه منفی انتخاب شده به‌طور تصادفی مثبت باشد.

در شکل (۱)، نمودار همگرایی برای الگوریتم علی بابا و چهل دزد مشخص شده است. بهترین هزینه به دست آمده از تابع هدف که از طریق الگوریتم علی بابا و چهل دزد به دست آمده، مشخص شده است. هزینه‌های به دست آمده در تعداد تکرارهای مختلف هستند. در الگوریتم علی بابا و چهل دزد، توانایی اکتشاف و بهره برداری با همگرایی دزدان به سمت راه حل بهینه سراسری تحقق می یابد. به طور دقیق، همگرایی به این معنی است که بیشتر دزدان در همان موقعیت در فضای جست‌وجو جمع می‌شوند. الگوریتم علی بابا و چهل دزد از چندین پارامتر استفاده می کند که منجر به اکتشاف و بهره برداری می شود. این پارامترها برای انجام فرآیند همگرایی الگوریتم علی بابا و چهل دزد مفید هستند. الگوریتم علی بابا و چهل دزد می‌تواند فضا را برای همه راه‌حل‌های ممکن برای شناسایی راه‌حل‌های بهینه یا غیربهینه بهتر جست‌وجو کند. دزدان، فضای جست‌وجو را در مکان‌ها و جهت‌های مختلف کاوش می‌کنند که نشان می‌دهد راه‌حل‌های بهتری ممکن است در مناطق امیدوارکننده دیگر پیدا شود.

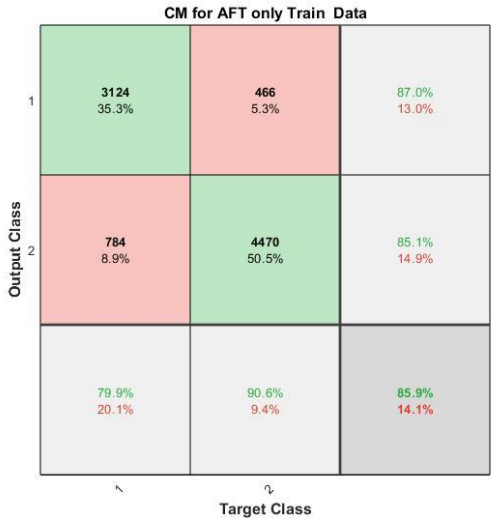


شکل (۱): نمودار همگرایی برای الگوریتم علی بابا و چهل دزد

در شکل (۲)، ماتریس درهم‌ریختگی برای الگوریتم علی بابا و چهل دزد در فاز آموزش و در شکل (۳)، ماتریس درهم‌ریختگی برای الگوریتم علی بابا و چهل دزد در فاز آزمایش مشخص شده است. همچنین در شکل (۴)، ماتریس درهم‌ریختگی برای روش پیشنهادی مبتنی بر الگوریتم علی بابا و چهل دزد مشخص شده است. با مقایسه این سه شکل می‌توان دریافت که روش پیشنهادی که از ترکیب شبکه عصبی پرسپترون چند لایه و الگوریتم علی بابا و چهل دزد استفاده کرده، نتایج بالاتری را کسب نموده است. ماتریس

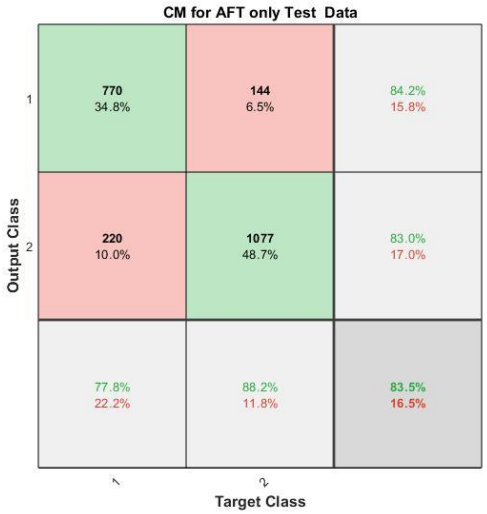
[1] Area Under Curve
[2] Receiver Operating Characteristic

درهم‌ریختگی در این سه شکل گواه بر ترکیب مناسب برای روش پیشنهادی و بهبود نتایج نسبت به زمانی است که به‌صورت تکی از الگوریتم‌ها استفاده شود. دلیل بهبود نتایج پرسپترون چند لایه از طریق الگوریتم علی بابا و چهل دزد این است که دو پارامتر مهم در الگوریتم علی بابا و چهل دزد وجود دارد که به آنها فاصله ردیابی و پتانسیل ادراک گفته می شود. با این دو پارامتر، الگوریتم علی بابا و چهل دزد می‌تواند فضا را برای همه راه‌حل‌های ممکن برای شناسایی راه‌حل‌های بهینه یا غیربهینه بهتر جست‌وجو کند. یکی دیگر از پارامترهای مهم در الگوریتم علی بابا و چهل دزد شبیه‌سازی روش های هوشمندانه مرجانه برای فریب دزدان است. به این ترتیب، دزدان فضای جست‌وجو را در مکان‌ها و جهت‌های مختلف کاوش می‌کنند که نشان می‌دهد راه‌حل‌های بهتری ممکن است در مناطق امیدوارکننده دیگر پیدا شود. بنابراین با پیدا کردن وزن‌های بهینه، روش پیشنهادی توانسته است نتایج را بهبود دهد.



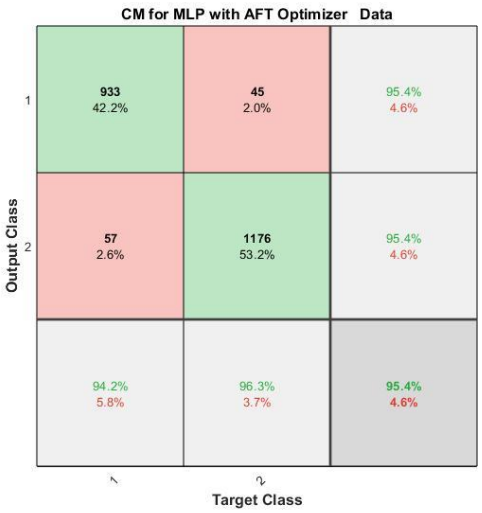
شکل (۲): ماتریس درهم‌ریختگی برای الگوریتم علی بابا و چهل دزد در فاز آموزش

[1] Tracking Distance
[2] Perception Potential

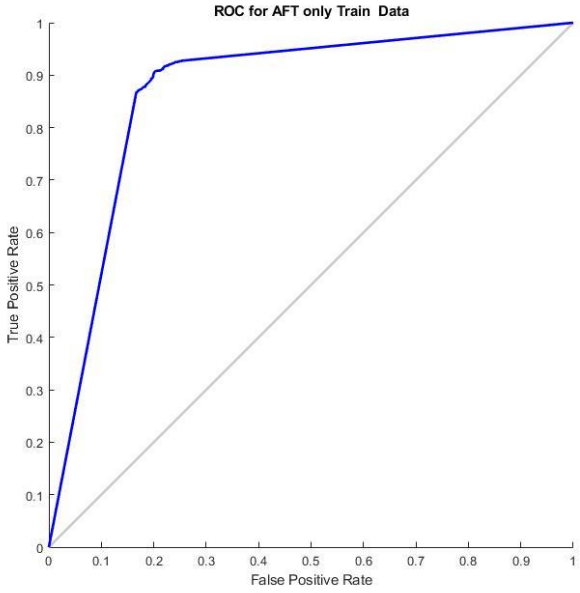


شکل (۳): ماتریس درهم‌ریختگی برای الگوریتم علی بابا و چهل دزد در فاز آزمایش

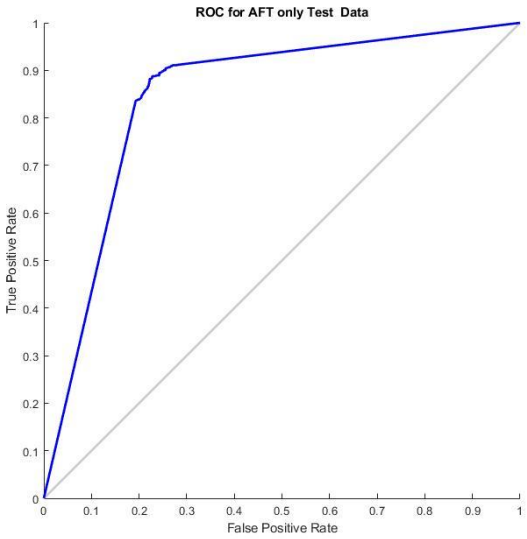
در شکل (۵)، منحنی ROC برای الگوریتم علی بابا و چهل دزد در فاز آموزش و در شکل (۶)، منحنی ROC برای الگوریتم علی بابا و چهل دزد در فاز آزمایش مشخص شده است. در شکل (۷)، منحنی ROC برای روش پیشنهادی مبتنی بر الگوریتم علی بابا و چهل دزد نشان داده شده است. منطقه زیر منحنی (AUC) اندازه‌گیری توانایی دسته‌بند برای تمایز بین کلاس‌ها است و به‌عنوان منحنی ROC استفاده می‌شود. هر چه AUC بالاتر باشد، عملکرد مدل در تشخیص کلاس‌های مثبت و منفی بهتر است؛ بنابراین طبق مطالب بیان شده، در منحنیROC ، مقدار بالاتر X نشان‌دهنده تعداد بیشتری از تشخیص‌های مثبت کاذب نسبت به نقاط منفی حقیقی است. درحالی‌که مقدار محور Y بالاتر نشان‌دهنده تعداد بیشتری از تشخیص‌های مثبت حقیقی نسبت به نقاط منفی کاذب است. با مقایسه نتایج سه شکل می‌توان دریافت که روش پیشنهادی که از ترکیب شبکه عصبی پرسپترون چند لایه و الگوریتم علی بابا و چهل دزد استفاده کرده است، نتایج ROC بالاتری را کسب نموده است. در این حالت، به این معنی است که روش پیشنهادی مبتنی بر الگوریتم علی بابا و چهل دزد کلاس نمونه را به درستی پیش‌بینی کرده است.



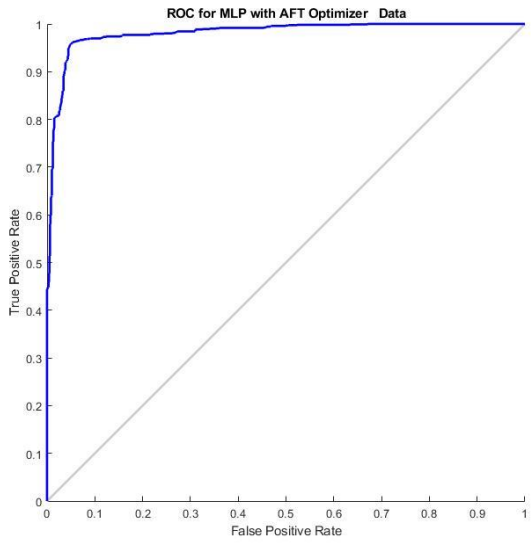
شکل (۴): ماتریس درهم‌ریختگی برای روش پیشنهادی مبتنی بر الگوریتم علی بابا و چهل دزد



شکل (۵): منحنی ROC برای الگوریتم علی بابا و چهل دزد در فاز آموزش



شکل (۶): منحنی ROC برای الگوریتم علی بابا و چهل دزد در فاز آزمایش



شکل (۷): منحنی ROC برای روش پیشنهادی مبتنی بر الگوریتم علی بابا و چهل دزد

در جدول (۲) نتایج از نظر معیار دقت با روش‌های ترکیبی شبکه عصبی و الگوریتم‌های فراابتکاری مقایسه می‌شود. دقت روش پیشنهادی که از ترکیب شبکه عصبی پرسپترون چند لایه و الگوریتم علی بابا و چهل دزد است، با دو رویکرد موجود در (Jalil و همکاران، ۲۰۲۲) مقایسه می‌شود. رویکرد موجود در (Jalil و همکاران، ۲۰۲۲) از الگوریتم پنگوئن امپراتور و الگوریتم ازدحام سالپ، برای بهینه‌سازی مدل آموزش دیده پیاده‌سازی شده‌اند.

جدول (۲): مقایسه از نظر دقت با روش‌های ترکیبی الگوریتم‌های فراابتکاری و شبکه عصبی

روش	دقت (درصد)
ANN – EPO (Jalil و همکاران، ۲۰۲۲)	۹۴,۰۹
ANN – SSA (Jalil و همکاران، ۲۰۲۲)	۹۳,۹۸
MLP- AFT (روش پیشنهادی)	۹۵,۳۹

در جدول (۳)، مقایسه نتایج روش پیشنهادی از نظر معیارهای دقت، صحت، F1_Score و AUC-ROC با الگوریتم‌های یادگیری ماشین موجود در پژوهش (Vijay و همکاران، ۲۰۲۲) انجام شده است. شش دسته‌بند مختلف مانند RF، LR، SVM، KNN، AdaBoost، MP (پرسپترون چند لایه)، J48 با روش پیشنهادی مقایسه شده اند. مطابق با نتایج به دست آمده، روش پیشنهادی توانسته است معیارهای ارزیابی را نسبت به الگوریتم‌های دیگر، بهبود دهد. الگوریتم علی بابا و چهل دزد دارای چندین مزیت متمایز بر اساس اصل اساسی خود است که باعث شده است نتایج به میزان مطلوبی بهبود یابد. برتری اول، مدل های به روز رسانی موقعیت از الگوریتم علی بابا و چهل دزد به طور موثر به افراد جمعیت کمک می کند تا هر منطقه را در فضای جست‌وجو اکتشاف و بهره‌برداری کنند. برتری دوم، جستجوی تصادفی که دزدان در فضای جست‌وجو استفاده می کنند، نه تنها تنوع جمعیت را افزایش می دهد، بلکه سرعت همگرایی را نیز تضمین می کندکه نشان دهنده تعادل کارآمد بین اکتشاف و بهره برداری است. برتری سوم، تعداد پارامترها در الگوریتم علی بابا و چهل دزد کم است، اما آنها توانایی خوبی برای بهبود قدرت و عملکرد آن دارند. برتری چهارم، بار محاسباتی الگوریتم علی بابا و چهل دزد کم است.

یکی از عیوب روش پیشنهادی، می تواند بار وارد شده به دلیل استفاده از دو الگوریتم باشد. به ویژه اینکه الگوریتم ها به صورت سری کار می کنند و الگوریتم علی بابا و چهل دزد مبتنی بر تکرار است.

جدول (۳): مقایسه نتایج روش پیشنهادی از نظر معیارهای مختلف

Accuracy (%)	ROC (%)	F1-score (%)	Precision (%)	دسته بند
۹۰,۴۸	۹۶,۵	۹۱,۵	۹۴,۳	LR
۸۹,۶۹	۹۰,۳	۹۰,۷	۹۵,۳	SVM
۹۱,۶۷	۹۳	۹۲,۸	۹۳,۴	KNN
۸۶,۳۶	۹۲,۴	۸۸	۹۰	AdaBoost
۹۱,۸۶	۹۷,۲	۹۲,۸	۹۵,۶	MP
۹۲,۹۳	۹۵,۴	۹۳,۸	۹۴,۸	J48
۹۴,۶۵	۹۸,۶	۹۵,۳	۹۶,۴	RF
۹۵,۳۹	۹۸,۱۲	۹۵,۸۴	۹۵,۳۹	روش پیشنهادی

۵- نتیجه گیری

با استفاده از یافتن وزن و بایاس‌های شبکه عصبی از طریق الگوریتم علی بابا و چهل دزد می‌توان صفحات فیشینگ را با دقت بالایی شناسایی کرد. در روش پیشنهادی برای دسته‌بندی و تشخیص حملات فیشینگ از شبکه عصبی پرسپترون چند لایه استفاده می‌شود. با داشتن مجموعه‌ای از وزن‌ها و مقدار بایاس شبکه عصبی پرسپترون، خروجی متناسب با داده‌های ورودی و وزن‌ها تولید می‌کند. وزن ها از طریق الگوریتم علی بابا و چهل دزد پیدا می‌شوند. تابع هدف در حالت کلی همان تابع هزینه‌ای است که قرار است الگوریتم علی بابا و چهل دزد آن را حل کند. دستاورد الگوریتم علی بابا و چهل دزد به‌عنوان متغیر، W ها (پارامترهای وزن‌ها و بایاس‌ها) هستند. هدف این است که الگوریتم علی بابا و چهل دزد W ها را رفته‌رفته بهینه کند؛ بنابراین ورودی، W هایی هستند که الگوریتم علی بابا و چهل دزد مشخص می‌کند. نکته مهم، انتخاب روشی است که تابع هزینه با آن محاسبه شود که شامل 'MSE'، 'RMSE' و 'Accuracy' هستند. مجموعه داده مورد استفاده از سایت https://archive.ics.uci.edu/ml/datasets/Website+Phishing است. در این مجموعه داده، ویژگی‌های مختلف مربوط به وب‌سایت‌های قانونی و فیشی را شناسایی و ۱۳۵۳ وب‌سایت مختلف را از منابع مختلف جمع‌آوری کرده‌اند. نتایج روش پیشنهادی با طرح پایه از نظر دقت، صحت، F1_Score و منحنی AUC-ROC مقایسه می‌شوند.

هدف مطالعه آینده، شناسایی وب‌سایت‌های مشکوک با افزودن لایه‌های بیشتر در شبکه عصبی و استفاده از شبکه‌های عصبی دقیق‌تر است و یک رویکرد مبتنی بر یادگیری عمیق برای شناسایی وب‌سایت‌های فیشینگ از طریق تجزیه‌وتحلیل URL ارائه خواهد شد. همچنین می‌توان از الگوریتم‌های فراابتکاری دیگر مانند بهینه‌سازی اسب برای یافتن وزن‌های شبکه عصبی استفاده کرد.

منابع

Abedin, N. F., Bawm, R., Sarwar, T., Saifuddin, M., Rahman, M. A., & Hossain, S. (2020). Phishing attack detection using machine learning classification techniques. In 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS) (pp. 1125-1130). IEEE.

Al-Andoli, M. N., Tan, S. C., Sim, K. S., Lim, C. P., & Goh, P. Y. (2022). Parallel Deep Learning with a hybrid BP-PSO framework for feature extraction and malware classification. Applied Soft Computing, 131, 109756.

Almseidin, M., Alkasassbeh, M., Alzubi, M., & Al-Sawwa, J. (2022). Cyber-Phishing Website Detection Using Fuzzy Rule Interpolation. Cryptography, 6(2), 24.

Altaher, A. (2017). Phishing websites classification using hybrid SVM and KNN approach. International Journal of Advanced Computer Science and Applications, 8(6).

Alzubi, O. A., Alzubi, J. A., Al-Zoubi, A. M., Hassonah, M. A., & Kose, U. (2022). An efficient malware detection approach with feature weighting based on Harris Hawks optimization. Cluster Computing, 25(4), 2369-2387.

Barreiro Herrera, D. A., & Camargo Mendoza, J. E. (2022). A Systematic Review on Phishing Detection: A Perspective Beyond a High Accuracy in Phishing Detection. In International Conference on Applied Informatics (pp. 173-188). Springer, Cham.

Bhagwat, S., & Gupta, G. P. (2022). Android Malware Detection Using Hybrid Meta-heuristic Feature Selection and Ensemble Learning Techniques. In International Conference on Advances in Computing and Data Sciences (pp. 145-156). Springer, Cham.

Braik, M., Ryalat, M. H., & Al-Zoubi, H. (2022). A novel meta-heuristic algorithm for solving numerical optimization problems: Ali Baba and the forty thieves. Neural Computing and Applications, 34(1), 409-455.

Darshan, S. S., Kumara, M. A., & Jaidhar, C. D. (2016). Windows malware detection based on cuckoo sandbox generated report using machine learning algorithm. In 2016 11th International Conference on Industrial and Information Systems (ICIIS) (pp. 534-539). IEEE.

Das, S., Nippert-Eng, C., & Camp, L. J. (2022). Evaluating user susceptibility to phishing attacks. Information & Computer Security.

Dhiyanesh, B., Selvanathan, N., Kiruthiga, G., & Radha, R. (2021). Effective attribute selection and classification technique for phishing attacks detection. In 2021 5th International Conference on Electronics, Communication and Aerospace Technology (ICECA) (pp. 1-7). IEEE.

Jafar, M. T., Al-Fawa'reh, M., Barhoush, M., & Alshira'H, M. H. (2022). Enhanced Analysis Approach to Detect Phishing Attacks During COVID-19 Crisis. Cybernetics and Information Technologies, 22(1), 60-76.

Jain, A. K., & Gupta, B. B. (2022). A survey of phishing attack techniques, defence mechanisms and open research challenges. Enterprise Information Systems, 16(4), 527-565.

Jalil, S., Usman, M., & Fong, A. (2022). Highly accurate phishing URL detection based on machine learning. Journal of Ambient Intelligence and Humanized Computing, 1-19.

Kovač, A., Dunđer, I., & Seljan, S. (2022). An overview of machine learning algorithms for detecting phishing attacks on electronic messaging services. In 2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO) (pp. 954-961). IEEE.

Minocha, S., & Singh, B. (2022). A novel phishing detection system using binary modified equilibrium optimizer for feature selection. Computers & Electrical Engineering, 98, 107689.

Palša, J., Ádám, N., Hurtuk, J., Chovancová, E., Madoš, B., Chovanec, M., & Kocan, S. (2022). MLMD—A Malware-Detecting Antivirus Tool Based on the XGBoost Machine Learning Algorithm. Applied Sciences, 12(13), 6672.

Ripa, S. P., Islam, F., & Arifuzzaman, M. (2021). The emergence threat of phishing attack and the detection techniques using machine learning models. In 2021 International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI) (pp. 1-6). IEEE.

Sabahno, M., & Safara, F. (2022). ISHO: improved spotted hyena optimization algorithm for phishing website detection. Multimedia Tools and Applications, 81(24), 34677-34696.

Stobbs, J., Issac, B., & Jacob, S. M. (2020). Phishing web page detection using optimised machine learning. In 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom) (pp. 483-490). IEEE.

Uplenchwar, S., Sawant, V., Surve, P., Deshpande, S., & Kelkar, S. (2022). Phishing Attack Detection on Text Messages Using Machine Learning Techniques. In 2022 IEEE Pune Section International Conference (PuneCon) (pp. 1-5). IEEE.

Vijay, J. S., Kulkarni, K., & Arya, A. (2022). Metaheuristic Optimization of Neural Networks for Phishing Detection. In 2022 3rd International Conference for Emerging Technology (INCET) (pp. 1-5). IEEE.

Zhang, X., Shi, D., Zhang, H., Liu, W., & Li, R. (2018). Efficient detection of phishing attacks with hybrid neural networks. In 2018 IEEE 18th International Conference on Communication Technology (ICCT) (pp. 844-848). IEEE.

Zhu, E., Yuan, Q., Chen, Z., Li, X., & Fang, X. (2022). CCBLA: a Lightweight Phishing Detection Model Based on CNN, BiLSTM, and Attention Mechanism. Cognitive Computation, 1-14.