ارائه یک سیستم تشخیص نفوذ جدید مبتنی‌بر الگوریتم تکامل تفاضلی و درخت تصمیم

محمد بشارتلو*[۱]، سمیرا مودّتی[۲]،

۱- موسسه آموزش عالی علوم و فناوری آریان، moohaamaad.beshaaratloo68@gmail.com

۲- استادیار دانشگاه مازندران، دانشکده فنی و مهندسی، دانشگاه مازندران ، بابلسر، ایران، s.mavaddati@umz.ac.ir

چکیده: از آنجاییکه داده مورد استفاده در سیستم تشخیص نفود حجم بالایی دارد، یکی از مسائل ضروری در این سیستم ها حفظ ویژگی‌های با بهترین کیفیت در مجموعه داده می‌باشد بطوریکه این ویژگی‌های منتخب قادر باشند، ساختار مجموعه داده‌ها را به درستی نشان دهند. بنابراین ضرورت خواهد داشت تا ویژگی‌های زائد و نامرتبط از مجموعه داده‌ها حذف و بهترین زیرمجموعه ویژگی از بین مجموع ویژگی‌ها تعیین گردد. در این مقاله، الگوریتم تکامل تفاضلی به عنوان استراتژی جستجو برای انتخاب بهترین زیرمجموعه ویژگی‌ها و طبقه‌بند درخت تصمیم برای تعیین کیفیت ویژگی‌های انتخاب شده مورد استفاده قرار می‌گیرد. مجموعه داده KDD Cup 99 برای ارزیابی روش پیشنهادی بکار برده می‌شود. نتایج شبیه‌سازی‌های انجام شده نشان می دهد که زیرمجموعه ویژگی بدست آمده توسط الگوریتم پیشنهادی به نرخ تشخیص و دقت بالاتر و نرخ هشدار نادرست پایین‌تر در مقایسه با نتایج بدست آمده با استفاده از الگوریتم کرم شب‌تاب و استفاده از تمام ویژگی‌ها دست می‌یابد.

کلمات کلیدی: سیستم تشخیص نفود، انتخاب ویژگی، الگوریتم تکامل تفاضلی، درخت تصمیم.

۱. مقدمه

به همراه رشد شبکه‌های کامپیوتری، حملات و نفوذها به این شبکه ها نیز گسترش یافته و به روش‌های مختلف صورت می‌پذیرد. این حملات و نفوذ را می‌توان به دو دسته نفودگرهای خارجی و داخلی دسته‌بندی کرد. نفودگرهای خارجی کسانی هستند که اجازه استفاده از سیستم را ندارند اما سعی می‌کنند که در سیستم راه یابند[۱]. در نقطه مقابل، نفودگرهای داخلی کسانی هستند که برای دستیابی به سیستم اختیارات محدودی دارند، اما تلاش می‌کنند به منابعی که اجازه استفاده از آنها را ندارند، دسترسی پیدا کنند. فرآیند نفود به مجموعه اقدامات غیرقانونی گفته می‌شود که صحت، محرمانگی و یا دسترسی یک منبع کامپیوتری را به خطر بیاندازد [۱].

رشد روز افزون استفاده از خدمات شبکه‌های کامپیوتری از یک سو و حمله به این شبکه‌ها و سیستم‌های کامپیوتری از سوی دیگر باعث شده است که تشخیص نفود به عنوان یک زمینه تحقیقاتی مهم در مساله امنیت این شبکه‌ها و مقابله با

* **Corresponding author**
Email: moohaamaad.beshaaratloo68@gmail.com

نفوذکنندگان به شبکه‌ها و سیستم‌های کامپیوتری مطرح شود. هدف از تشخیص نفوذ این است که استفاده غیرمجاز، سواستفاده و آسیب رساندن به سیستم‌ها و شبکه‌های کامپیوتری توسط نفوذگران شناسایی شود. بنابراین، سیستم تشخیص نفوذ ابزار امنیتی موثر و کارآمدی است که در شبکه‌های کامپیوتری قرار می‌گیرد و با استفاده از یک سری قوانین از پیش تعریف شده به دسترسی کاربران نظارت می‌نماید و تصمیم می‌گیرد که این فعالیت‌ها براساس یکپارچگی، قابلیت اطمینان و دسترس‌پذیری منابع اطلاعات، مخرب یا طبیعی هستند [۲]. در نتیجه، این سیستم‌ها برای محافظت از سیستم‌های کامپیوتری از خطرهای نفوذ و اختلال بکار برده می‌شوند [۳]. دو روش اصلی به نام‌های روش‌های تشخیص misuse و anomaly برای تشخیص نفود وجود دارد [۴، ۵]. هیچکدام از این دو روش برای تشخیص تمام نوع‌های نفوذ و حمله‌ها موثر نیستند و هر یک از آنها توانایی‌ها و ضعف‌هایی دارند [۶]. روش misuse برای شناسایی حمله‌های معروف موثر است اما برای حمله‌های دیده نشده موثر نمی‌باشد [۷]. در مقابل، روش تشخیص anomaly برای تشخیص حمله‌های جدید موثر می‌باشد اما آن مشکل نرخ هشدار نادرست بالا را دارد [۸]. به همین دلیل تعدادی از سیستم‌های تشخیص نفوذ روش ترکیبی را بکار می‌برند که هر دو تکنیک misuse و anormaly را یکپارچه می‌نمایند [۴].

اکثر تحقیقات در زمینه تشخیص نفوذ بر روی تکنیک anomaly متمرکز شده‌اند زیرا آن امیدوارکننده‌تر به نظر می رسد [۹، ۱۰]. اخیرا، روش‌های زیادی در زمینه تشخیص anomaly شامل تکنیک‌های داده‌کاوی، یادگیری ماشین، تحلیل آماری و هوش مصنوعی ارائه شده‌اند [۱۱]. در [۱۲] چندین تکنیک آماری برای سیستم‌های تشخیص anomaly شامل اندازه‌گیری حدآستانه، میانگین، انحراف استاندارد و مدل‌های چندمتغیره بکار برده شد. در میان آنها، تکنیک‌های داده‌کاوی و یادگیری ماشین که طبقه‌بندهای منفرد [۱۳، ۱۴] و طبقه‌بندهای ترکیبی [۱۵، ۱۶] را شامل می‌شوند بطور گسترده‌ای برای حل تعداد زیادی از مسائل دسته‌بندی سیستم تشخیص نفوذ بکار برده می‌شوند. علاوه بر این، الگوریتم‌های هوش محاسباتی شامل الگوریتم‌های ژنتیک، سیستم ایمنی مصنوعی و هوش جمعی به عنوان ابزاری برای حل مسائل تشخیص نفود بکار برده شدند [۱۷-۱۹].

به علت بعد بالای داده‌ها، بکار بردن این تکنیک‌ها بویژه هنگامی که برای تشخیص نفوذ بصورت آنی جستجو می‌کنند، شدیدا زمان‌بر است [۱۹]. به همین دلیل، انتخاب ویژگی که بخشی از کاهش ابعاد است برای انتخاب زیرمجموعه بهینه ویژگی‌ها به منظور نمایش کل داده‌ها نیاز است [۲۰]. بنابراین، اندازه داده‌ها کاهش و دقت افزایش می‌یابد زیرا حذف موثر ویژگی‌های نامرتبط و افزونه از مجموعه داده اصلی توسط انتخاب ویژگی اثبات شده است [۱۹].

در این مقاله، یک روش انتخاب ویژگی برای سیستم تشخیص نفوذ پیشنهاد می‌شود تا زیرمجموعه بهینه از ویژگی‌ها را تولید نماید. روش پیشنهادی براساس الگوریتم تکامل تفاضلی برای جستجوی زیرمجموعه بهینه ویژگی‌ها است. همچنین، درخت تصمیم به عنوان طبقه‌بند بکار برده می‌شود تا کیفیت زیرمجموعه‌های ویژگی تولید شده را بهبود دهد.

ادامه متن مقاله به این صورت سازماندهی شده است: قسمت ۲ معرفی مختصری از درخت تصمیم و الگوریتم تکامل تفاضلی را ارائه می‌نماید. روش انتخاب ویژگی پیشنهادی براساس الگوریتم تکامل تفاضلی و درخت تصمیم در قسمت ۳ توضیح داده می‌شود. نتایج آزمایشها در قسمت ۴ گزارش می‌شود. در نهایت، در قسمت ۵ نتیجه‌گیری و پیشنهادات برای کارهای آتی بیان می‌شود.

۲. معرفی درخت تصمیم و الگوریتم تفاضل تکاملی

در این بخش ابتدا طبقه بند درخت تصمیم به طور مختصر معرفی می گردد و سپس الگوریتم تکامل تفاضلی شرح داده می شود.

۱.۲. درخت تصمیم

درخت تصمیم یکی از معروفترین تکنیک‌های یادگیری ماشین است که توسط Quinlan معرفی شد[۲۱]. درخت تصمیم سه مولفه اصلی شامل گره‌ها، یال‌ها و برگ‌ها را دارد. هر گره فضای نمونه را به دو زیرفضا یا بیشتر براساس تابع گسسته ساز از مقادیر ویژگی ورودی جدا می نماید. گره ریشه همچنین گره تست نامیده می شود که هیچ یال ورودی ندارد. هر یال خروجی از گره با یک مقدار ویژگی برچسب دهی می شود و هر برگ با یک دسته یا کلاس برچسب گذاری می شود. درخت در طول فاز آموزش با کمک داده‌های آموزشی ساخته می شود. در فاز تست، هر نمونه از داده‌های تست توسط جهت یابی پایین از ریشه درخت به سمت یک برگ براساس خروجی داده تست در طول مسیر طبقه بندی می شود. دو الگوریتم محبوب ID3 و C4.5 برای ساخت درخت تصمیم بکار برده می شوند [۲۱]. در این مقاله، از طبقه‌بند مبتنی‌بر درخت تصمیم ID3 برای تعیین کیفیت ویژگی‌های انتخاب شده استفاده می‌شود.

۲.۲. الگوریتم تکامل تفاضلی

الگوریتم تکامل تفاضلی (DE)[1] یک الگوریتم تکاملی قدرتمند ساده است که توسط Price و Storn پیشنهاد گردید [۲۲]. الگوریتم DE یک روش جستجوی تصادفی مبتنی بر جمعیت برای مسائل بهینه سازی سراسری در فضای جستجوی پیوسته است. الگوریتم DE از عملیات‌هایی مانند تقاطع، جهش و انتخاب روی جمعیت استفاده می نماید تا تابع هدف را نسبت به نسل‌ها بهینه نماید. الگوریتم DE از گام‌های زیر تشکیل شده است:

- گام۱: جمعیت اولیه در فضای جستجو با توزیع یکنواخت بطور تصافی ایجاد می شود.

$$x_i(j) = lb_j + (ub_j - lb_j) \times rand, \;\; j = 1, \dots, d \; and \; i = 1, \dots, N \qquad (۱)$$

که rand تابعیست که یک عدد تصادفی در بازه $[0,1]$ تولید می نماید. lb_j و ub_j به ترتیب کران پایین و کران بالای بعد j–ام از فضای جستجو هستند. d تعداد کل ابعاد راه حل است و N تعداد کل افراد موجود در جمعیت است. سپس مقدار برازندگی هر راه حل محاسبه می شود.

- گام۲: عملگر جهش بر روی هر بردار x_i (فرد از جمعیت) اجرا می گردد تا بردار v_i را برای آن بدست آورد. ۵ نمونه از پرکاربردترین استراتژی‌های جهش بکار برده شده عبارتند از:

- DE/rand/1

$$v_i = x_{r_1} + F * (x_{r_2} - x_{r_3}) \qquad (۲)$$

- DE/best/1

$$v_i = x_{best} + F * (x_{r_1} - x_{r_2}) \qquad (۳)$$

- DE/current-to-best/1

[1] **Differential evolution (DE)**

$$v_i = x_i + F * (x_{best} - x_i) + F * \left(x_{r_1} - x_{r_2}\right) \quad (۴)$$

- DE/best/2

$$v_i = x_{best} + F * \left(x_{r_1} - x_{r_2}\right) + F * \left(x_3 - x_{r_4}\right) \quad (۵)$$

- DE/rand/2

$$v_i = x_{r_5} + F * \left(x_{r_1} - x_{r_2}\right) + F * \left(x_3 - x_{r_4}\right) \quad (۶)$$

که r_1، r_2، r_3، r_4 و r_5 اعداد صحیح تصادفی انتخاب شده از $\{1,2,3,\dots,N\}$ است و مقادیرشان با i فرق دارند. بردار x_{best} بهترین فرد با بهترین برازش از جمعیت است. فاکتور مقیاس گذاری F عدد حقیقی مثبت برای مقیاس گذاری بردارهای متفاوت است.

- گام۳: عملگر تقاطع بر روی هر جفت بردار x_i و v_i با رابطه (۷) اجرا می شود تا بردار u_i را بدست آورد.

$$u_i(j) = \begin{cases} v_i(j) & if\ (rand \le CR) or\ (j = j_{rand}) \\ x_i(j) & otherwise \end{cases} \quad (۷)$$

که $i = 1,2,\dots,N$ ،$j = 1,2,..,d$ و نرخ تقاطع[1] عدد ثابتی بین ۰ تا ۱ است که توسط توسط کاربر مشخص می‌شود تا کسر مقادیر پارامترهای کپی شده از بردار v_i را کنترل نماید. j_{rand} عدد صحیح تصادفی انتخاب شده از ۱ تا d است تا تضمین نماید که u_i حداقل یک مولفه از v_i را بدست می آورد.

- گام ۴: عملگر انتخاب دو بردار x_i و u_i را برحسب مقدار تابع هدف مقایسه می نماید تا برداری را که برای نسل بعد زنده می ماند تعیین نماید. عملگر انتخاب براساس رابطه (۸) خواهد بود. در صورتیکه تابع هدف بصورت ماکزیمم سازی تعریف شود آنگاه برداری که دارای مقدار تابع هدف بزرگتری است در جمعیت نسل بعد قرار داده می‌شود.

$$x_i = \begin{cases} u_i & ,if\ f(u_i) \ge f(x_i) \\ x_i & ,\ otherwise \end{cases} \quad (۸)$$

در این رابطه، $f(.)$ مقدار تابع هدف را به ازای ورودیش باز می‌گرداند.

- گام ۵: متغیر شمارنده نسل افزایش می یابد و در صورتیکه به ماکزیمم نسل‌ها دست نیابد به گام ۲ بازمی گردد و در غیر اینصورت اجرای الگوریتم خاتمه می یابد.

[1] **Crossover rate**

۳. انتخاب ویژگی براساس الگوریتم تکامل تفاضلی

الگوریتم تکامل تفاضلی برای فضای پیوسته تعریف شده است. برای اینکه این الگوریتم قادر باشد تا با مسائل باینری سروکار داشته باشد، پس از اجرای عملگر جهش و تقاطع بر روی هر فرد از جمعیت، باید راه حل x_i حاصل بصورت باینری تبدیل شود که برای اینکار رابطه (۹) بکار برده می شود [۲۳]:

$$x_i(j) = \begin{cases} 1 & ,if\ rand \le \exp(-\exp(-|x_i(j)|)) \\ 0 & ,otherwise \end{cases} \quad (۹)$$

نسخه باینری الگوریتم DE از گام‌های زیر تشکیل شده است:

- گام۱: جمعیت اولیه در فضای جستجو با توزیع یکنواخت بطور تصافی ایجاد می گردد:

$$x_i(j) = round(rand),\ \ j = 1, \dots, d\ and\ i = 1, \dots, N$$

که $rand$ تابعی است که یک عدد تصادفی در بازه $[0,1]$ تولید می نماید. d تعداد کل ابعاد راه حل است و N تعداد کل اعضای جمعیت است. هر راه حل از جمعیت به صورت باینری مقداردهی اولیه می شود. سپس مقدار برازندگی هر فرد از جمعیت براساس زیرمجموعه ویژگیهای انتخاب شده توسط هر راه حل محاسبه می شود. برای اینکار، با کمک زیرمجموعه ویژگیهای انتخابی راه حل و مجموعه دادههای آموزشی طبقه بند درخت تصمیم ساخته می شود و سپس مقدار برازندگی آن راه حل با کمک تابع برازندگی تعریف شده محاسبه می‌شود.

- گام۲: عملگر جهش نوع DE/best/1 طبق رابطه (۳) بر روی هر بردار x_i (فرد از جمعیت) اجرا می شود تا بردار $\mathrm{v_i}$ را برای آن بدست آید.

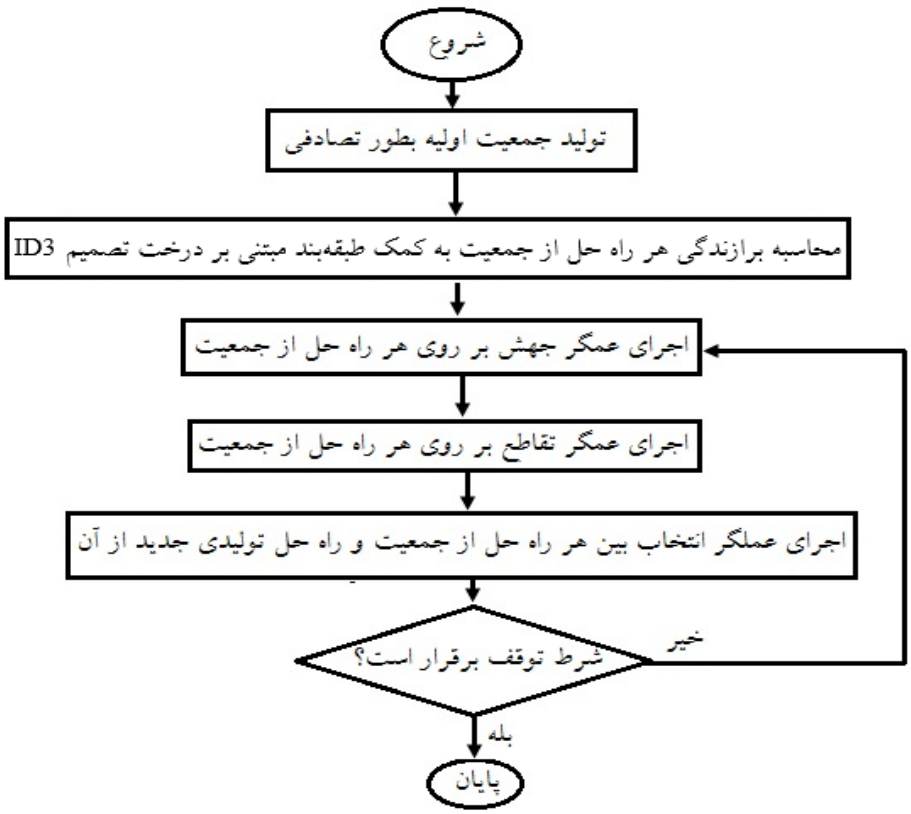
- گام۳: عملگر تقاطع بر روی هر جفت بردار $\mathrm{x_i}$ و $\mathrm{v_i}$ طبق رابطه (۷) اجرا می گردد تا بردار u_i را تولید نماید.

- گام۴: عمل گسسته سازی بر روی بردار u_i با رابطه (۹) اجرا می شود.

- گام ۵: عملگر انتخاب، دو بردار $\mathrm{x_i}$ و $\mathrm{u_i}$ را برحسب مقدار تابع هدف مقایسه می نماید تا برداری را که برای نسل بعد زنده می ماند تعیین نماید که عملگر انتخاب با رابطه (۸) است (در صورتیکه تابع هدف بصورت ماکزیمم سازی تعریف شود آنگاه برداری که دارای مقدار تابع هدف بزرگتری است در جمعیت نسل بعد قرار داده می شود).

گام ۶: متغیر شمارنده نسل افزایش داده می شود و در صورتیکه به ماکزیمم نسلها دست نیابد به گام ۲ بازمی گردد و در غیر اینصورت اجرای الگوریتم خاتمه می یابد.

بلوک دیاگرام روش پیشنهادی در شکل (۱) نشان داده شده است.



شکل۱. بلوک دیاگرام روش پیشنهادی.

۱.۳. تابع برازندگی

سه معیار برای ارزیابی کارایی سیستم تشخیص نفوذ بکار برده می شود که عبارتند از: ۱- نرخ تشخیص حمله[۱] (ADR) ۲-نرخ مثبت اشتباه[۲] یا هشدار نادرست (FPR) ۳-دقت سیستم[۳] (SA) یا نرخ دقت[۴] (AR) [۲۴]. هر یک از این معیارها به ترتیب با روابط (۱۰)، (۱۱) و (۱۲) تعریف می شوند.

$$DR = \frac{\text{تعداد حملاتی که به درستی حمله تشخیص داده شده‌اند}}{\text{تعداد کل حملات در داده تست}} \times 100\% \qquad (۱۰)$$

[۱] **Attack detection rate**
[۲] **False positive rate**
[۳] **System accuracy**
[۴] **Accuracy rate**

$$FPR = \frac{\text{تعداد نفوذهای نرمال که به اشتباه حمله تشخیص داده شده‌اند}}{\text{تعداد کل نفوذهای نرمال در داده تست}} \times 100\% \quad (۱۱)$$

$$AR = \frac{\text{تعداد نمونه‌های درست تشخیص داده شده}}{\text{کل نمونه‌های داده تست}} \times 100\% \quad (۱۲)$$

هر سیستم تشخیص نفوذ می‌بایست نرخ تشخیص حمله را بهبود دهد و نرخ هشدار نادرست را کاهش دهد. بنابراین مقادیر بالاتر DR و AR و مقادیر کمتر برای FPR عملکرد دسته‌بندی بهتری را برای سیستم‌های تشخیص نفوذ نشان می دهد.

برازندگی تعریف شده برای تابع بهینه‌سازی مورد استفاده، میزان شانس انتخاب هر راه حل را برای حضور در نسل بعد تعیین می نماید. در این مقاله، تابع برازندگی براساس دو معیار DR و FPR تعریف می‌شود. بنابراین راه حلی که بیشترین مقدار DR و کمترین مقدار FPR را بدست دهد، بالاترین مقدار برازندگی را خواهد داشت. ترکیب این دو معیار تابع برازندگی را بصورت (۱۳) تعریف می نماید:

$$Fitness = \alpha * DR + \beta * (1 - FPR) \quad (۱۳)$$

این معادله بیان می نماید که DR و FPR اهمیت متفاوتی براساس دو ضریب α و β دارند که $\alpha \in [0,1]$ و $\beta = 1 - \alpha$ است. این پارامترها در طول روال شبیه‌سازی به صورت $\alpha = 0.7$ و $\square = 0.3$ درنظر گرفته شده‌اند. برای محاسبه برازندگی هر فرد از جمعیت ابتدا براساس زیرمجموعه ویژگیهای انتخاب شده توسط آن فرد درخت تصمیم با الگوریتم ID3 ساخته می شود و سپس هر داده تست براساس درخت ساخته شده طبقه بندی می شود و براساس برچسب‌های کلاسی بدست آمده برای داده‌ها، مقدار برازندگی طبق رابطه (۱۳) محاسبه می شود.

۴. نتایج شبیه‌سازی

در شبیه‌سازی‌های انجام شده از مجموعه داده KDD Cup 99 برای تعیین میزان کارایی الگوریتم پیشنهادی استفاده می‌گردد[۲۵]. رکوردهای موجود در این مجموعه داده دارای ۴۱ ویژگی به اضافه یک برچسب کلاس می باشد. رکوردها شامل ۲۱ نوع حمله می باشند بنابراین رکوردهای بصورت نرمال یا یکی از انواع حمله برچسب گذاری می شوند.

چهار دسته حمله به شرح زیر در مجموعه داده KDD Cup 99 مورد بررسی قرار می‌گیرد:

- حمله DoS[1]: منابع سیستم در این نوع حمله بیش از حد مورد استفاده قرار می‌گیرد و موجب رد شدن درخواست‌های نرمال برای در اختیار گرفتن منابع می‌شود.

[1] **Denial of service**

- حمله R2L: حمله کننده در این نوع حمله با نفوذ به ماشین قربانی به صورت غیرمجاز و از راه دور از طریق حدس زدن رمز عبور، از حساب قانونی کاربر سوء استفاده کرده و اقدام به ارسال بسته بر روی شبکه می‌نماید.

- حمله U2R: این نوع حمله ها به طور موفقیت آمیزی در ماشین قربانی اجرا شده و حمله کننده دسترسی های کاربر ارشد محلی را در اختیار می‌گیرد.

- حمله Probing: سیستم‌ها در این نوع از حمله‌ها، به منظور جمع‌آوری اطلاعات و یا یافتن قابلیت‌های آسیب‌پذیری نظارت و کاوش می شوند.

مجموعه داده‌های آموزشی و تست از KDD Cup 99 به ترتیب شامل ۴۹۴۰۲۰ و ۳۱۱۰۲۸ نمونه می باشد. همانطور که مشاهده می‌شود ابعاد این مجموعه داده‌ها برای استفاده بسیار بزرگ است. به همین دلیل دو زیرمجموعه داده (آموزشی و تست) بطور تصادفی از آن‌ها استخراج می شود و برای حفظ نسبت هر نوع حمله در هر دو مجموعه آموزشی و تست، تعداد نمونه‌های هر حمله بر ۱۰۰ تقسیم می‌گردد. برای مثال، تعداد حمله‌های ipsweep در مجموعه داده‌های آموزشی و تست اصلی ۱۲۴۷ و ۳۰۶ می باشد در حالیکه تعداد آنها در مجموعه داده‌های استخراج شده ۱۲ و ۳ می باشد. جدول ۱ انواع مختلف حمله و تعداد رخداد متناظرشان را در داده‌های آموزشی و تست به ترتیب نشان می دهد. در آزمایش‌های انجام شده تعداد داده‌های آموزشی ۴۹۴۷ و تعداد داده‌های تست ۳۱۱۷ می باشند که بطور تصادفی انتخاب شده‌اند. از جدول ۱، probing(41;42) به معنی آن است که تعداد رکوردها از حمله Prob در مجموعه آموزشی ۴۱ است در حالیکه تعداد رکوردها از این نوع حمله در مجموعه تست برابر ۴۲ می باشد.

جدول۱. انواع مختلف حمله و تعداد رخداد متناظرشان به ترتیب در مجموعه داده آموزشی و تست.

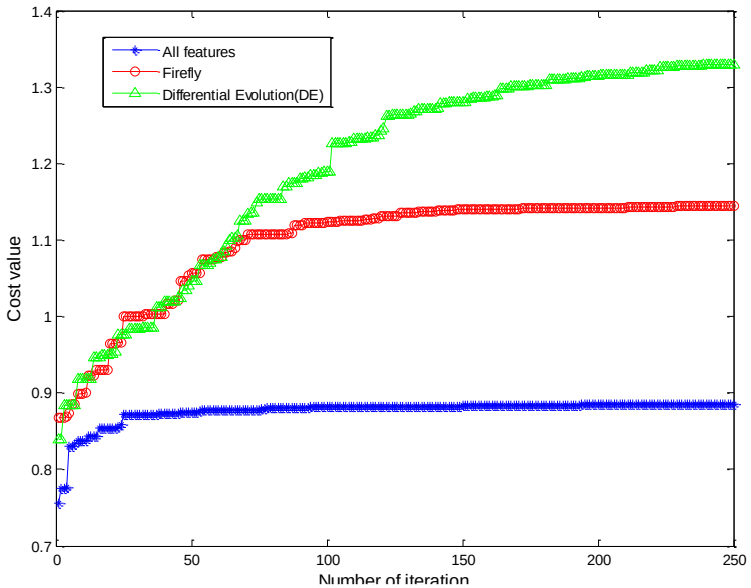
Normal(973;606)			
Probing (41; 42)	DoS(3915; 2299)	U2R(5; 10)	R2L(13; 160)
ipsweep(12;3), Mscan(0;11), Nmap(2;1) Portsweep(11;4) Saint(0;7), Satan(16;16).	apache2(0;8), back(22;11), land(0; 0), mailbomb(0;50), Neptune(1072;580), processtable(0;8), Pod(3;1), udpstorm(0;0), Smurf(2808;1641), Teardrop(10;0),	buffer_overflow(3;1), httptunnel(0;3), loadmodule(0;0), perl(0;0), rootkit(2;2), xterm(0;2), Ps(0;2), Sqlattack(0;0),	ftp_write(0;0), imap(0;0), guesspasswd(2;44), named(0;0), multihop(0;0), phf(0;0), sendmail(0;0), snmpgetattack(0;77), snmpguess(0;24), spy(0;0,) warezclient(10;0), worm(0;0), warezmaster(1;15), xsnoop(0;0), xlock(0;0),

روش پیشنهادی در آزمایش‌های انجام شده که بر مبنای الگوریتم بهینه‌سازی تکامل تفاضلی است، با الگوریتم کرم شب‌تاب و حالت استفاده از تمام ویژگی‌ها مقایسه می‌شود. اندازه جمعیت اولیه در هریک از الگوریتم‌ها ۳۰ و ماکزیمم تعداد تکرار الگوریتم‌ها ۵۰ می باشد. آزمایش ۲۰ مرتبه بصورت مستقل اجرا می شود و مقادیر میانگین معیارهای ارزیابی سیستم تشخیص برای این ۲۰ اجرای مستقل گزارش می شود. نتایج بدست آمده از آزمایش در جدول ۲ گزارش شده است.

جدول۲- نتایج سه معیار ارزیابی سیستم تشخیص نفوذ برای روشهای مختلف.

روش	DR(%)	AR(%)	FPR(%)
الگوریتم کرم شب‌تاب	92.138	91.446	0.664
الگوریتم تکامل تفاضلی	**92.674**	**92.244**	**0.662**
درخت تصمیم با تمام ویژگی‌ها	71.087	73.267	17.685

همانطور که در جدول ۲ مشاهده می‌شود روش پیشنهادی مبتنی بر الگوریتم بهینه‌سازی تکامل تفاضلی از نظر هر سه معیار DR، نرخ دقت (AR) و FPR نسبت به روش تشخیص نفوذ مبتنی‌بر الگوریتم کرم شب‌تاب و استفاده از تمام ویژگی‌ها با استفاده از الگوریتم تکامل تفاضلی به بهترین نتایج دست یافته است. موفقیت روش پیشنهادی مبتنی‌بر تکامل تفضلی در انتخاب زیرمجموعه ویژگی بهینه نسبت به روش‌های دیگر خواهد بود. همچنین منحنی همگرایی روش‌های تشخیص نفوذ مبتنی‌بر الگوریتم بهینه‌سازی تکامل تفاضلی، کرم شب و درخت تصمیم با استفاده از تمام ویژگی‌ها در شکل ۲ نمایش داده شده است. در این شکل همگرایی بهترین جواب برای تابع هزینه پیشنهادی براساس الگوریتم بهینه‌سازی تکامل تفاضلی بدست آمده است. همانطور که قابل مشاهده است، میانگین راه‌حل‌ها در الگوریتم تکامل تفاضلی به راه‌حل بهینه در هر تکرار بسیار نزدیک بوده که نشان می‌دهد جمعیت به درستی در فضای مسئله به جستجوی راه‌حل‌ها پرداخته‌اند.



شکل ۲- منحنی همگرایی حل مسئله تشخیص نفوذ توسط الگوریتم‌های مختلف.

۵. نتیجه‌گیری

در این مقاله، روش جدیدی مبتنی بر ترکیب الگوریتم تکامل تفاضلی و درخت تصمیم به منظور انتخاب ویژگی‌های بهینه مورد نیاز برای اجرای روال تشخیص نفوذ در سیستم‌های کامپیوتری ارئه گردید. داده مورد نیاز برای فرآیند آموزش و تست از مجموعه KDD Cup 99 انتخاب گردید. در ابتدا، روال اصلاح الگوریتم تکامل تفاضلی به منظور تطبیق با شرایط موجود در فرآیند انتخاب ویژگی ارائه گردید. سپس طبقه‌بند مبتنی‌بر درخت تصمیم برای ارزیابی ویژگی‌های انتخاب شده در الگوریتم تکامل تفاضلی باینری بکار برده شد. نتایج شبیه‌سازی‌های انجام شده نشان داد که زیرمجموعه ویژگی‌های بهینه انتخابی توسط الگوریتم بهینه‌سازی تکامل تفاضلی پیشنهادی بیشترین مقدار معیارهای DR و AR و کمترین مقدار معیار FPR را بدست می‌دهد. همچنین در کارهای آینده، بررسی استفاده از تکنیک‌های دیگر برای طبقه‌بندی مانند ماشین بردار پشتیبان، شبکه‌های عصبی و روش‌های خوشه بندی پیشنهاد می شود.

مراجع

[1] Zamboni, D. (2001), “Using internal sensors for computer intrusion detection,” Center for Education and Research in Information Assurance and Security, Purdue University, 2001.

[2] Chung, Y.Y. and Wahid, N. (2012), “A hybrid network intrusion detection system using simplified swarm optimization (SSO),” Applied Soft Computing, 12, pp. 3014-3022.

[3] Liao, H.J., Lin, C.H.R., Lin, Y.C. and Tung, K.Y. (2013), “Intrusion detection system: A comprehensive review,” Journal of Network and Computer Applications, 36, pp. 16-24.

[4] Depren, O., Topallar, M., Anarim, E. and Ciliz, M.K. (2005), “An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks,” Expert systems with Applications, 29, pp. 713-722.

[5] Wang, G., Hao, J., Ma, J. and Huang, L. (2010), “A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering,” Expert systems with applications, 37, pp. 6225-6232.

[6] Lin, W.C., Ke, S.W. and Tsai, C.F. (2015), “CANN: An intrusion detection system based on combining cluster centers and nearest neighbors,” Knowledge-based systems, 78, pp. 13-21.

[7] Zhang, J., Li, H., Gao, Q., Wang, H. and Luo, Y. (2015) “Detecting anomalies from big network traffic data using an adaptive detection approach,” Information Sciences, 318, pp. 91-110.

[8] Kim, G., Lee, S. and Kim, S. (2014), “A novel hybrid intrusion detection method integrating anomaly detection with misuse detection,” Expert Systems with Applications, 41, pp. 1690-1700.

[9] Gan, X.S., Duanmu, J.S., Wang, J.f. and Cong, W. (2013), "Anomaly intrusion detection based on PLS feature extraction and core vector machine," Knowledge-Based Systems, 40, pp. 1-6.

[10] Karami, A. and Guerrero-Zapata, M. (2015), "A fuzzy anomaly detection system based on hybrid pso-kmeans algorithm in content-centric networks," Neurocomputing, 149, pp. 1253-1269.

[11] Rastegari, S., Hingston, P. and Lam, C.P. (2015), "Evolving statistical rulesets for network intrusion detection," Applied Soft Computing, 33, pp. 348-359.

[12] Denning, D.E. (1987), "An intrusion-detection model," IEEE Transactions on software engineering, pp. 222-232.

[13] Li, Y. and Guo, L., "An active learning based TCM-KNN algorithm for supervised network intrusion detection," Computers & security, 26, pp. 459-467, 2007.

[14] Chen, W.H., Hsu, S.H. and Shen, H.P. (2005), "Application of SVM and ANN for intrusion detection," Computers & Operations Research, 32, pp. 2617-2634.

[15] Zainal, A., Maarof, M.A. and Shamsuddin, S.M. (2009), "Ensemble classifiers for network intrusion detection system," Journal of Information Assurance and Security, 4, pp. 217-225.

[16] Mukkamala, S., Sung, A.H. and Abraham, A. (2005), "Intrusion detection using an ensemble of intelligent paradigms," Journal of network and computer applications, 28, pp. 167-182.

[17] Wu, S.X. and Banzhaf, W. (2010), "The use of computational intelligence in intrusion detection systems: A review," Applied Soft Computing, 10, pp. 1-35.

[18] Shafi, K. and Abbass, H.A. (2007), "Biologically-inspired complex adaptive systems approaches to network intrusion detection," Information Security Technical Report, 12, pp. 209-217.

[19] Khammassi, C. and Krichen, S. (2017), "A GA-LR Wrapper Approach for Feature Selection in Network Intrusion Detection," Computers & Security, 70, pp.255-277.

[20] Eesa, A.S., Orman, Z. and Brifcani, A.M.A. (2015), "A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems," Expert Systems with Applications, 42, pp. 2670-2679.

[21] Salzberg, S. (1994), "Book Review: C4. 5: Programs for machine learning by J," Ross Quinlan, pp. 1-6.

[22] Storn, R. and Price, K. (1997), "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," Journal of global optimization, vol. 11, pp. 341-359.

[23] He, X., Zhang, Q., Sun, N. and Dong, Y. (2009), "Feature selection with discrete binary differential evolution," International Conference on Artificial Intelligence and Computational Intelligence, pp. 327-330.

[24] Chen, R.C., Cheng, K.F., Chen, Y.H. and Hsieh, C.F. (2009), "Using rough set and support vector machine for network intrusion detection system," in Intelligent Information and Database Systems, First Asian Conference on, pp. 465-470.

[25] http://kdd.ics.uci.edu