Computer Communications 202 (2023) 191–203

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

تشخیص ناهنجاری برای تشخیص خطا در شبکه‌های اجتماعی بی‌سیم با استفاده از یادگیری ماشین

Llorenç Cerdà-Alabern [a,*], Gabriel Iuhasz [b], Gabriele Gemmi [a,c]

[a] Universitat Politècnica de Catalunya, Barcelona, Spain
[b] West University, Timisoara, Romania
[c] University of Venice Ca' Foscari, Italy

اطلاعات مقاله

واژه‌های کلیدی:
تشخیص خطا
تشخیص ناهنجاری
یادگیری ماشین
مجموعه‌داده شبکه بی‌سیم
شبکه‌های اجتماعی بی‌سیم

چکیده

یادگیری ماشین در سال‌های اخیر توجه روزافزونی را در علوم کامپیوتر به خود جلب کرده و انواع بسیاری از روش‌ها پیشنهاد شده‌اند[10]. در شبکه‌های کامپیوتری، توجه کمی به استفاده از یادگیری ماشین (ML) برای تشخیص خطا شده است [11]، که دلیل اصلی آن کمبود مجموعه‌داده (dataset) است[12]. این امر ناشی از عدم تمایل اپراتورهای شبکه برای اشتراک‌گذاری داده‌های مربوط به زیرساخت‌ها و خرابی‌های شبکه خود است[13]. در این مقاله، ما تلاش می‌کنیم با استفاده از تکنیک‌های تشخیص ناهنجاری، این شکاف را پر کنیم تا رویدادهای خرابی سخت‌افزاری را در شبکه‌های اجتماعی بی‌سیم تشخیص دهیم[14]. برای این منظور، ما از ۴ رویکرد یادگیری ماشین بدون نظارت (unsupervised) استفاده می‌کنیم که بر اساس اصول متفاوتی بنا شده‌اند[15]. ما یک مجموعه‌داده از یک شبکه اجتماعی بی‌سیم در حال کار (production) ساخته‌ایم که شامل ویژگی‌های ترافیکی و غیر ترافیکی، مانند بار پردازنده (CPU) و حافظه است[16]. برای تحلیل عددی، ما توانایی رویکردهای مختلف ML را در تشخیص یک خرابی ناخواسته درگاه (gateway) که در طول جمع‌آوری داده رخ داد، بررسی کردیم[17]. نتایج عددی ما نشان می‌دهد که وقتی ویژگی‌های غیر ترافیکی نیز در نظر گرفته می‌شوند، عملکرد همه رویکردهای آزمایش‌شده در تشخیص خرابی درگاه بهبود می‌یابد[18]. ما می‌بینیم که وقتی به درستی تنظیم شوند، همه روش‌های ML در تشخیص خرابی مؤثر هستند[19]. با این وجود، با استفاده از مرزهای تصمیم‌گیری و سایر تکنیک‌های تحلیلی، ما تفاوت‌های رفتاری قابل توجهی را در میان روش‌های ML مشاهده می‌کنیم[20].

1. مقدمه

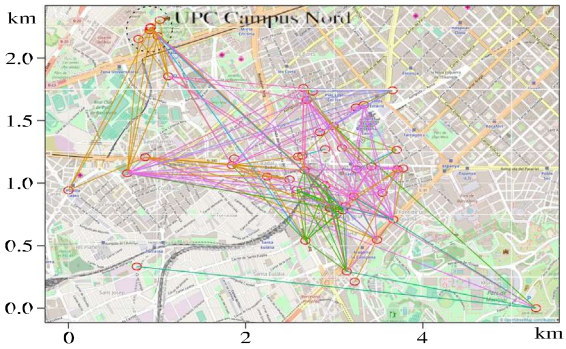
تشخیص ناهنجاری (AD) با هدف شناسایی انحرافات از رفتار مورد انتظار انجام می‌شود[22]. ثابت شده است که شناسایی این رفتارهای غیرمعمول، ابزاری قدرتمند در طیف گسترده‌ای از علوم کاربردی است[23]. برخی نمونه‌ها شامل تشخیص کلاهبرداری کارت اعتباری، تشخیص پزشکی، فرآیندهای صنعتی و شبکه‌های کامپیوتری است[24]. اگرچه روش‌های مختلفی را می‌توان برای AD به کار برد [25]، پیشرفت‌های اخیر در روش‌های ML و توانایی آن‌ها در یادگیری از داده‌ها، تعداد پیشنهادات AD با استفاده از تکنیک‌های ML را افزایش داده است[26]. یادگیری ماشین از یک مجموعه‌داده برای پیش‌بینی‌های احتمالی استفاده می‌کند[27]. گروهی از نمونه‌های مجموعه‌داده که به عنوان «مجموعه آموزش» شناخته می‌شوند، برای آموزش الگوریتم استفاده می‌شوند[28]. سپس، پیش‌بینی‌ها بر روی گروه متفاوتی از نمونه‌ها که «مجموعه آزمایشی» نامیده می‌شوند، انجام می‌پذیرد[29]. در برخی کاربردها، مجموعه آموزش برچسب‌گذاری شده است[30]. به عنوان مثال، در تشخیص نوری کاراکترها، مجموعه آموزش شامل تصاویر و برچسب‌های متناظر با کاراکترهای آن‌هاست[31]. این مسائل، «یادگیری با نظارت» نامیده می‌شوند[32]. در مقابل، در AD معمولاً از یک مجموعه‌داده بدون برچسب استفاده می‌شود و الگوریتم‌های ML، «بدون نظارت» نامیده می‌شوند[33]. در AD، مجموعه آموزش صرفاً به عنوان نمایشی از عملیات عادی مورد انتظار استفاده می‌شود و ناهنجاری‌ها با انحراف از رفتار مورد انتظار شناسایی می‌شوند[34].

عمدتاً به امنیت، به‌ویژه تشخیص نفوذ به شبکه، معطوف بوده است[35]. در مقابل، ما در این ADدر حوزه شبکه‌های کامپیوتری، مقاله بر روی «تشخیص خطا» تمرکز می‌کنیم [3636]، یکی دیگر از نیازهای اساسی در شبکه‌های کامپیوتری می‌تواند در آن بسیار مورد توجه باشد، اما توجه کمی به آن شده است. برخی مشکلات، کارهای اندکی را که می‌توان در ادبیات موضوع در مورد AD برای تشخیص خطا یافت، توضیح می‌دهند. مشکل اصلی، کمبود مجموعه‌داده است. دلیل آن این است که یک مجموعه‌داده واقع‌بینانه برای تشخیص خطا باید شامل ویژگی‌های مرتبط با ترافیک شبکه و معیارهای سخت‌افزاری، مانند بار پردازنده و استفاده از حافظه باشد[40]. تولید این نوع مجموعه‌داده از طریق شبیه‌سازی دشوار خواهد بود، بلکه باید از یک بستر آزمایشی واقعی یا در حالت ایده‌آل، از یک شبکه در حال کار به دست آید[41]. با این حال، به دلایل محرمانگی، اپراتورهای شبکه تجاری این نوع مجموعه‌داده‌ها را از شبکه‌های خود عمومی نمی‌کنند[42].

بنابراین، هدف دیگر کار ما ایجاد یک مجموعه‌داده با داده‌های واقعی است[43]. برای انجام این کار، ما بر روی AD در شبکه‌های اجتماعی بی‌سیم (WCN) تمرکز خواهیم کرد[44]. شبکه‌سازی اجتماعی، که به عنوان شبکه‌سازی از پایین به بالا نیز شناخته می‌شود، مدلی است که از نیاز به دسترسی گسترده به اینترنت در مناطق محروم، معمولاً مناطق روستایی و کشورهای در حال توسعه، پدید آمده است[45]. امروزه صدها شبکه اجتماعی وجود دارند که به روش‌های بسیار متنوعی فعالیت می‌کنند[46]. WCNها شبکه‌های غیرانتفاعی هستند که توسط کاربران خودشان ساخته می‌شوند، معمولاً با نصب آنتن‌های بی‌سیم بر روی سقف خانه‌هایشان[47]. گاهی اوقات کاربران WCN نه تنها زیرساخت شبکه را می‌سازند، بلکه برای تشکیل ISP های خرد (micro ISP) با هم متحد می‌شوند و دسترسی به اینترنت و خدمات داخلی مدیریت‌شده توسط جامعه را فراهم می‌کنند[48]. WCN ها شبیه به ارائه‌دهندگان خدمات اینترنت بی‌سیم (WISP) هستند که معمولاً

* Corresponding author.
E-mail address: llorenc.cerda@upc.edu (L. Cerdà-Alabern).

https://doi.org/10.1016/j.comcom.2023.02.019
Received 20 September 2022; Received in revised form 18 January 2023; Accepted 20 February 2023
Available online 23 February 2023



ig . 1/. توپولوژی GuifiSants. رنگ‌ها نشان‌دهنده‌ی لینک‌هایی هستند که در یک کانال WiFi یکسان پیکربندی شده‌اند و روترهای بیرونی را نمایش می‌دهند.

- تا جایی که ما اطلاع داریم، این اولین مجموعه‌داده‌ای است که شامل چنین تنوع غنی از ویژگی‌های به‌دست‌آمده از یک شبکه بی‌سیم در حال کار است که در یک مخزن عمومی موجود است[79].
- ما از روش‌های AD با استفاده از این مجموعه‌داده برای انجام تحلیل تشخیص خطا با استفاده از ۴ رویکرد یادگیری بدون نظارت ML مبتنی بر اصول مختلف استفاده کرده‌ایم[80]. ما عملکرد این روش‌های شناخته‌شده ML را که برای AD در شبکه‌های بی‌سیم به کار می‌روند و رفتار آن‌ها را با تغییر برخی پارامترهای سیستم، مانند اندازه مجموعه‌داده، بررسی می‌کنیم[81].

ادامه مقاله به این صورت سازمان‌دهی شده است: بخش ۲ برخی از کارهای مرتبط را ارائه می‌دهد[82]. بخش ۳، ۴ رویکرد ML مورد استفاده در این مقاله را خلاصه می‌کند[83]. بخش ۳ متدولوژی و ویژگی‌های جمع‌آوری‌شده در مجموعه‌داده مورد مطالعه را توصیف می‌کند[84]. بخش ۵ نتایج عددی را ارائه می‌دهد [85]و بخش ۶ مقاله را به پایان می‌رساند[86].

2. کارهای مرتبط

تشخیص ناهنجاری به مسئله یافتن الگوهایی در داده‌ها اشاره دارد که با رفتار مورد انتظار مطابقت ندارند[88]. در آمار، اغلب از اصطلاح «داده پرت» برای این مفهوم استفاده می‌شود[89]. اهمیت AD به این دلیل است که ناهنجاری‌ها در داده‌ها ممکن است نشان‌دهنده رفتارهای نادرست حیاتی در سناریوهای متعدد باشند[90]. به عنوان مثال، تشخیص کلاهبرداری برای کارت‌های اعتباری، بیمه یا مراقبت‌های بهداشتی، تشخیص آسیب و غیره[91]. اهمیت این موضوع منجر به تعداد زیادی مقاله، بررسی و حتی کتاب شده است[92]. به عنوان مثال، بررسی‌های [1, 20] مروری گسترده بر تکنیک‌های تشخیصی ارائه می‌دهند که در حوزه‌های متعددی توسعه یافته‌اند[93].

در شبکه‌های کامپیوتری، بیشتر کارهای مربوط به AD به طور سنتی بر روی مسئله امنیت متمرکز بوده‌اند[94]. در این زمینه، اغلب از اصطلاح «سیستم تشخیص نفوذ» استفاده می‌شود[95]. IDSها معمولاً از ناهنجاری‌ها در ترافیک شبکه برای تشخیص طیف گسترده‌ای از حملات امنیتی، مانند حملات محروم‌سازی از سرویس ، شناسایی و غیره بهره می‌برند[96]. مروری بر IDSهای پیشنهادی در ادبیات موضوع را می‌توان در [7] یافت[97]. در [22]، PCA برای انجام تحلیل ساختاری ترافیک شبکه معرفی شد[98]. پیشنهاد آن‌ها توجه قابل توجهی را برای IDS به خود جلب کرد[99]. به عنوان مثال، در [23، 24] اصلاحاتی در تحلیل PCA برای تشخیص جهش‌ها در جریان‌های ترافیکی انجام شده است، که به طور بالقوه حملات شبکه را شناسایی می‌کند [100]. PCA برای تشخیص ناهنجاری‌های ترافیک شبکه توسط تعدادی از مقالات نیز مورد انتقاد قرار گرفته است[101]. در [27] ضعف‌های PCA تحلیل شده و از Commute Distance برای تشخیص حملات DoS با استفاده از اندازه‌گیری‌های ترافیک استفاده شده است[102]. در [14] استدلال می‌شود که مشکلات PCA ناشی از نقص در پذیرش آن است و روش‌های مناسبی برای اعمال PCA پیشنهاد می‌شود[103].

روش‌های ML سابقه طولانی در استفاده برای AD دارند[104]. تا همین اواخر، بیشتر تحقیقات بر روی استفاده از روش‌های ML با نظارت همراه با مجموعه‌داده‌های مصنوعی یا به شدت حاشیه‌نویسی‌شده انجام می‌شد[105]. اخیراً این روند به سمت استفاده از مجموعه‌داده‌های واقعی همراه با روش‌های بدون نظارت تغییر کرده است[106]. با این حال، مقالات تحقیقاتی زیادی وجود ندارند که هم مقایسه‌ای از روش‌های AD بدون نظارت و هم یک مجموعه‌داده واقعی در دسترس عموم را داشته باشند.

مرور بسیار خوبی از روش‌های بدون نظارت بر روی چندین مجموعه‌داده در [29] آورده شده است[108]. در اینجا ما عملکرد چندین روش شناخته‌شده AD مانند k-NN، CBLOF و غیره را می‌بینیم[109]. تحقیقات مرتبط‌تری شامل برخی از همین روش‌ها در [30] انجام شده است که در آن نویسندگان چندین تکنیک AD مبتنی بر خوشه‌بندی را بر روی داده‌های ترافیک شبکه مقایسه می‌کنند[110]. انواع دیگر روش‌های AD نیز در سیستم‌های مدیریت شبکه با داده‌های شبکه با ابعاد بالا استفاده شده‌اند[111].

کسب‌وکارهای کوچکی با حاشیه سود کاهش‌یافته هستند و هدفشان ارائه اینترنت گسترده به جامعه خود است[50].

ماهیت غیرانتفاعی و منابع مشترک شبکه‌های اجتماعی معمولاً کاربران آن‌ها را برای مشارکت در پروژه‌های تحقیقاتی و فراهم کردن دسترسی به داده‌های زیرساخت شبکه خود، پذیرا می‌سازد [51]. بنابراین ما با هدف ساخت یک مجموعه‌داده از یک شبکه در حال کار، بر روی WCN تمرکز کرده‌ایم[52]. به طور خاص، ما Guifi.net را در نظر گرفته‌ایم[53]. Guifi.net یکی از نمونه‌های برجسته WCN است[54]. این شبکه در سال ۲۰۰۴ آغاز به کار کرد و در زمان نگارش این مقاله، ۳۶۰۸۸۶ گره فعال را گزارش می‌دهد[55]. Guifi.net به یک شبکه پیچیده تبدیل شده است که در آن انجمن‌های کاربران خود-تأمین‌کننده با اپراتورهای شبکه تجاری همزیستی دارند[56].

در این مقاله، ما یک مجموعه‌داده را که از یک WCN در حال کار که بخشی از Guifi.net است جمع‌آوری کرده‌ایم، تحلیل می‌کنیم[57]. این WCN در محله‌ای از بارسلون، اسپانیا، مستقر شده است و GuifiSants نامیده می‌شود [58]. هنگامی که مجموعه‌داده جمع‌آوری شد، حدود ۶۰ گره وجود داشت[59]. GuifiSants از نظر توپولوژی، کیفیت پیوندها و رفتار گره‌ها بسیار ناهمگن است[60]. گره‌هایی در ساختمان‌های بلندتر وجود دارند که دارای چندین آنتن هستند، برخی سکتور و برخی دیگر پارابولیک که پیوندهای نقطه به نقطه با ظرفیت بالا ایجاد می‌کنند[61]. این گره‌ها نوعی ستون فقرات (backbone) برنامه‌ریزی‌نشده را تشکیل می‌دهند[62]. سایر کاربران غیر فنی دارای گره‌های نهایی هستند که از یک آنتن واحد ساخته شده‌اند[63]. شکل ۱ مکان‌های جغرافیایی گره‌های GuifiSants و تصویری از روترهای فضای باز مستقر در یک گره را نشان می‌دهد[64646464]. بسته به نوع آنتن‌ها، انسداد پیوندها و فناوری‌های WiFi 802.11n/ac مورد استفاده، ظرفیت پیوند از چند مگابیت بر ثانیه تا چند صد مگابیت بر ثانیه متغیر است[65]. پایداری گره‌ها نیز بسیار متغیر است[66]. به عنوان مثال، کاربرانی هستند که مرتباً نرم‌افزار را به‌روزرسانی می‌کنند یا اگر اتصال رضایت‌بخش نباشد، گره خود را راه‌اندازی مجدد می‌کنند[67]. گره‌های ستون فقرات پایدارتر هستند و خرابی یا راه‌اندازی مجدد کمتری دارند[68]. از آنجایی که این یک شبکه مِش است، اتصال نسبتاً انعطاف‌پذیر است: حتی اگر یک گره ستون فقرات از کار بیفتد، پروتکل مسیریابی مسیرهای جایگزین را مجدداً پیکربندی می‌کند و اکثر گره‌ها همچنان می‌توانند به اینترنت، که سرویس اصلی مصرفی کاربران است، دسترسی داشته باشند[69].

AD در چنین WCN به دلیل ماهیت ناهمگن و متنوع آن چالش‌برانگیز است[70]. با این وجود، ما معتقدیم که این نه تنها یک مطالعه جالب به خودی خود است، بلکه سناریوهای دیگری مانند شبکه‌های بزرگ اینترنت اشیاء نیز ممکن است ویژگی‌های تنوع مشابهی داشته باشند، که به ما امکان می‌دهد برخی از درس‌های آموخته‌شده از تحلیل WCN را تعمیم دهیم[71].

اکثر مطالعات AD مرتبط با شبکه‌های کامپیوتری که در ادبیات موضوع یافت می‌شوند، از ویژگی‌های مرتبط با ترافیک استفاده می‌کنند[72]. در مقابل، مجموعه‌داده‌ای که ما تولید کرده‌ایم نه تنها شامل ترافیک، بلکه شامل ویژگی‌هایی از هسته لینوکس است که به عنوان مثال، وضعیت پردازنده و حافظه را منعکس می‌کند [73]. چنین ویژگی‌هایی معمولاً توسط ابزارهای نظارت بر شبکه مانند Munin و Nagios جمع‌آوری می‌شوند [74]. با این حال، این ابزارها صرفاً گرافیک‌های سری زمانی از ویژگی‌های جمع‌آوری‌شده تولید می‌کنند و تحلیل و تفسیر آن‌ها به مدیر شبکه واگذار می‌شود[75]. بنابراین ما علاقه‌مندیم بدانیم که آیا گنجاندن این ویژگی‌های اضافی به ویژگی‌های اساسی مبتنی بر ترافیک می‌تواند قابلیت AD را افزایش دهد یا خیر[76].

به طور خلاصه، مشارکت‌های اصلی ما در این مقاله به شرح زیر است:

- ما یک مجموعه‌داده تولید کرده‌ایم که مجموعه بزرگی از ویژگی‌های مرتبط نه تنها با ترافیک، بلکه با پارامترهای دیگری مانند پردازنده و حافظه را جمع‌آوری می‌کند[77]. این مجموعه‌داده از یک WCN در حال کار جمع‌آوری شده و ما آن را در مخزن عمومی Zenodo در دسترس قرار داده‌ایم.

روش‌های مبتنی بر یادگیری عمیق نیز برای AD استفاده می‌شوند. در [32] نویسندگان مروری جامع بر روش‌های مختلف یادگیری عمیق برای AD شامل روش‌های بدون نظارت ارائه می‌دهند[113]. آن‌ها همچنین به درستی اشاره می‌کنند که «توضیح‌پذیری» یک مسئله کلیدی است، به‌ویژه هنگام استفاده از روش‌های AD که به طور سنتی به عنوان مدل‌های جعبه-سیاه (black-box) در نظر گرفته می‌شوند[114]. مسئله توضیح‌پذیری همچنین در [33, 34] در مورد مدل‌های مختلف یادگیری عمیق، که شامل خودرمزگذارها [35]و خودرمزگذارهای متغیر [36] هستند، مورد بحث قرار گرفته است[115]. بیشتر تحقیقات انجام‌شده در مورد روش‌های بدون نظارت برای AD ، عمدتاً بر روی تشخیص نفوذ به شبکه متمرکز است[116]. [37] یکی از معدود کارهای یافت‌شده در ادبیات موضوع است که در آن AD شبکه با استفاده از ML با یک مجموعه‌داده واقعی بررسی شده است[117]. با این حال، در [37] نویسندگان از AD برای مطالعه رفتار ردیابی‌های TCPجهت بررسی عملکرد یک شبکه سلولی G4 استفاده می‌کنند که سناریوی کاملاً متفاوتی نسبت به ماست[118].

روش‌های مورد استفاده در ادبیات موضوع را می‌توان به چند نوع تقسیم کرد[119]. اینها شامل روش‌های مبتنی بر خوشه‌بندی [38-41]، تکنیک‌های مبتنی بر داده پرت [37, 42-45] و تکنیک‌های مبتنی بر محاسبات نرم [46-48] است[120]. همه روش‌های توصیف‌شده در اینجا علاوه بر مشکل توضیح‌پذیری که قبلاً ذکر شد، دارای مزایا و معایبی هستند[121]. روش‌های مبتنی بر خوشه‌بندی معمولاً فقط می‌توانند ویژگی‌های پیوسته را مدیریت کنند؛ معیارهای نزدیکی نامناسب بر نرخ تشخیص تأثیر منفی می‌گذارد[122]. بیشتر روش‌های مبتنی بر داده پرت پیچیده هستند و به شدت به پارامترها وابسته‌اند، در حالی که روش‌های مبتنی بر محاسبات نرم معمولاً به مقادیر زیادی داده تاریخی نیاز دارند و مشکل «بیش‌برازش» مدل‌ها مسلماً مسئله بزرگ‌تری نسبت به روش‌های با نظارت است[123]. در اصل، انتخاب اینکه از چه نوع روشی و چه پارامترهای آموزشی استفاده شود، بسیار به حوزه مسئله و داده‌های موجود بستگی دارد[124].

3. رویکردهای مبتنی بر یادگیری ماشین

در بخش بعدی، ما بر روی آزمایش چندین تکنیک ML بدون نظارت برای AD تمرکز خواهیم کرد[126]. برای تکنیک‌های ML، AD بر اساس نوع روش‌ها و ویژگی‌های داده‌های موجود به چند دسته تقسیم می‌شود[127]. ساده‌ترین شکل ناهنجاری‌ها، ناهنجاری‌های نقطه‌ای هستند که می‌توانند تنها با یک ویژگی مشخص شوند و تشخیص آن‌ها آسان‌تر است[128]. انواع دیگر ناهنجاری‌ها پیچیده‌تر هستند اما در نهایت درک بسیار عمیق‌تری از عملکردهای درونی یک سیستم و/یا برنامه تحت نظارت ارائه می‌دهند[129]. این نوع ناهنجاری‌ها در سیستم‌های پیچیده توزیع‌شده جغرافیایی بسیار رایج هستند[130].

ناهنجاری‌های زمینه‌ای در مورد سیستم‌های پیچیده بسیار جالب هستند[131]. این نوع ناهنجاری‌ها زمانی اتفاق می‌افتند که الگوی خاصی از مقادیر ویژگی‌ها مشاهده می‌شود[132]. به صورت مجزا، این مقادیر ناهنجار نیستند، اما وقتی در زمینه مشاهده می‌شوند، یک رویداد ناهنجار را نشان می‌دهند[133]. این نوع ناهنجاری‌ها می‌توانند نشان‌دهنده تنگناهای برنامه، خرابی قریب‌الوقوع سخت‌افزار، پیکربندی نادرست نرم‌افزار یا حتی فعالیت مخرب باشند[134]. آخرین انواع عمده ناهنجاری‌ها که مرتبط هستند، ناهنجاری‌های زمانی و ترتیبی هستند که در آن‌ها یک رویداد خاص خارج از ترتیب یا در زمان‌های نادرست رخ می‌دهد[135]. این نوع ناهنجاری‌ها در سیستم‌هایی که رابطه فضایی-زمانی قوی بین ویژگی‌ها دارند، بسیار مهم هستند، که این مورد در سیستم‌های توزیع‌شده پویا مانند شبکه‌های مِش بسیار صادق است[136].

3.1. تحلیل مؤلفه‌های اصلی

این واقعیت است که داده‌های باکیفیت و برچسب‌گذاری‌شده به ندرت ADیک ملاحظه مهم برای اکثر وظایف در دسترس هستند[137]. علاوه بر این، روش‌های با نظارت که بر روی داده‌های برچسب‌گذاری‌شده آموزش دیده‌اند، قادر به تشخیص رویدادهای ناهنجار جدید یا پیش‌بینی‌نشده نخواهند بود[138]. روش‌های بدون نظارت این محدودیت را ندارند، با این حال، این روش‌ها دارای چندین نقطه ضعف هستند[139]. اولاً، آنها در بسیاری از موارد مستعد نرخ بالای «مثبت کاذب» هستند[140]. این می‌تواند به دلایل زیادی مانند بیش‌برازش، فضاهای ویژگی بزرگ و غیره باشد[141]. ثانیاً، حتی اگر ناهنجاری‌ها تشخیص داده شوند، خود الگوریتم‌ها معمولاً قادر به ارائه هیچ بینش معناداری در مورد اینکه چه نوع ناهنجاری تشخیص داده شده یا حتی چه چیزی باعث رویداد ناهنجار شده است، نیستند[142]. در این بخش از مقاله، ما بر روی کاهش برخی از مسائل ارائه‌شده در اینجا در مورد روش‌های تشخیص بدون نظارت تمرکز خواهیم کرد[143]. توجه داشته باشید که ما هر اندازه‌گیری از مجموعه‌داده را به عنوان «نمونه» ارجاع خواهیم داد[144]. بنابراین، ما از اصطلاحات رویداد ناهنجار، نمونه ناهنجار یا به سادگی ناهنجاری، به جای یکدیگر استفاده خواهیم کرد[145].

تحلیل مؤلفه‌های اصلی یک متدولوژی استاندارد است که در کنترل فرآیند آماری به کار می‌رود[147]. ایده PCA تقریب زدن نمونه‌های n ویژگی با یک تصویر بر روی یک فضای با ابعاد کمتر l < n است[148]. PCA به خوبی در ادبیات موضوع مطالعه شده است، بنابراین ما از توصیف این روش صرف‌نظر می‌کنیم و فقط در مورد آماره‌هایی که برای تحلیل خود استفاده کردیم بحث می‌کنیم[149]. خواننده علاقه‌مند می‌تواند برای جزئیات بیشتر به [14] مراجعه کند[150]. به منظور تشخیص ناهنجاری‌های یک امتیاز جدید، چندین شاخص وجود دارد[151]. ما دریافتیم که در مطالعه ما، بهترین عملکرد مربوط به خطای پیش‌بینی مربع (SPE) است که به عنوان آماره Q نیز شناخته می‌شود:

$$Q_j = \sum_{i=1}^{n} e_{ji}^2 \quad (1)$$

که در آن e_{ji} باقیمانده i-ام ویژگی‌های نمونه j است.

هنگامی که یک ناهنجاری تشخیص داده می‌شود، به یک سیستم تشخیص برای تعیین علل ریشه‌ای آن نیاز است. رویکرد کلی استفاده از «نمودارهای مشارکت» است. ایده چنین نمودارهایی، تخمین مشارکت هر ویژگی مشاهده‌شده از یک نمونه در یک مقدار آماری خاص است که ناهنجار در نظر گرفته شده است. روش‌های مختلفی برای ساخت نمودارهای مشارکت پیشنهاد شده است. در این مقاله، ما از «مشارکت‌های تجزیه کامل» (CDC) استفاده خواهیم کرد. محاسبه CDC آسان است و عملکرد تشخیص خوبی را در یک سیستم نظارت بر شبکه نشان می‌دهد. CDC برای آماره Q که در بالا تعریف شد، توسط [49] داده شده است:

$$CDC_i^{Q_j} = e_{ji}^2. \quad (2)$$

Isolation Forest

در سال‌های اخیر، آشکارسازهای مبتنی بر گروه در مورد AD محبوبیت پیدا کرده‌اند[165]. این روش‌ها خروجی‌های چندین الگوریتم به نام «آشکارسازهای پایه» را ترکیب می‌کنند که سپس برای ایجاد یک خروجی یکپارچه استفاده می‌شوند[166]. این الگوریتم‌ها بر این فرض عمل می‌کنند که برخی الگوریتم‌ها بر روی زیرمجموعه‌ای از داده‌های موجود به خوبی عمل می‌کنند در حالی که برخی دیگر می‌توانند بر روی زیرمجموعه متفاوتی بهتر عمل کنند[167]. از سوی دیگر، ترکیب گروهی اغلب بهتر از برآوردگرهای منفرد عمل می‌کند، به دلیل توانایی آن‌ها در ترکیب خروجی‌های چندین الگوریتم[168]. بیشتر روش‌های گروهی به داده‌های برچسب‌گذاری‌شده نیاز دارند، با این حال، هم از دیدگاه نظری [51] و هم عملی [52] نشان داده شده است که روش‌های گروهی بدون نظارت ویژگی‌هایی با همتایان با نظارت خود دارند[169]. ما می‌توانیم یک موازنه بایاس-واریانس اصلاح‌شده را برای تنظیم تحلیل ناهنجاری فرموله کنیم [51]، این امر بسیاری از الگوریتم‌های با نظارت را قادر می‌سازد تا به وظایف بدون نظارت «تعمیم» یابند[170]. یکی از مسائل کلیدی که نتیجه مستقیم این تعمیم است، افزایش دشواری در انتخاب «فراپارامترهای» مناسب است[171].

جنگل ایزوله‌سازی یک الگوریتم مبتنی بر گروه داده پرت است که از چندین «درخت ایزوله‌سازی» ساخته شده است[172]. این الگوریتم زیرفضاهای تصادفی از داده‌ها را کاوش می‌کند[173]. در اصل، این الگوریتم زیرفضاهای محلی تصادفی را کاوش می‌کند زیرا هر درخت از تقسیم‌های متفاوتی استفاده می‌کند[174]. امتیازدهی با تعیین اینکه یافتن یک زیرفضای محلی با ابعاد کم که در آن یک رویداد خاص ایزوله شده است چقدر آسان است، انجام می‌شود[175]. به عبارت دیگر، فاصله از برگ تا ریشه به عنوان امتیاز داده پرت استفاده می‌شود[176]. مشابه جنگل تصادفی که یک روش با نظارت است

خودرمزگذارهای متغیر

رمزگذارهای خودبازسازی متغیر یا مدل‌هایی از شبکه‌های عصبی عمیق هستند که برای آموزش بدون‌ناظر طراحی شده‌اند و می‌توان از آن‌ها در وظایف تشخیص ناهنجاری (AD) استفاده کرد [57،58]. این مدل‌ها اغلب همراه با رمزگذارهای خودبازسازی معمولی یا (AE) مطرح می‌شوند که آن‌ها نیز مدل‌های یادگیری عمیق با اجزای توپولوژیکی مشابهی مانند رمزگذار و بازرمزگذار هستند. رمزگذار تلاش می‌کند تا نمایش پایین‌بعدی از داده‌های ورودی را بیاموزد مشابه تحلیل مؤلفه‌های اصلی یا (PCA و بازرمزگذار سعی دارد داده‌های ورودی را در بُعد اصلی بازسازی کند)AE ها معمولاً ساختاری متقارن دارند .هدف AEها این است که داده‌ها را به‌گونه‌ای رمزگذاری کنند که خطای بازسازی کاهش یابد. در کاربردهای تشخیص ناهنجاری، خطای بازسازی AE می‌تواند به‌عنوان امتیاز ناهنجاری مورد استفاده قرار گیرد، مشروط بر اینکه AE داده‌های آموزشی کافی برای دستیابی به حداقل خطای بازسازی برای داده‌های نرمال داشته باشد [59]

در مقابل، VAEها تلاش می‌کنند داده‌ها را به یک توزیع پنهان چندمتغیره رمزگذاری کنند که سپس توسط بازرمزگذار مورد استفاده قرار می‌گیرد. تفاوت اصلی بین AE و VAE در این است که به‌جای تولید یک بردار پنهان که بازرمزگذار بتواند آن را بازسازی کند، VAEها دو بردار را می‌آموزند که پارامترهای میانگین و واریانس یک توزیع را نشان می‌دهند؛ از این توزیع، بردار پنهان نمونه‌گیری شده و توسط تابع بازرمزگذار برای بازسازی ورودی اصلی استفاده می‌شود. به‌طور کلی، فضای پنهان تولیدشده توسط VAEها به سمت نرمال بودن گرایش دارد، که بخش عمده‌ای از آن به دلیل منظم‌سازی شدید رمزگذارها از طریق جمله واگرایی کولبک–لایبلر (Kullback–Leibler divergence) است.

VAEها برخی چالش‌های رایج با سایر مدل‌های یادگیری عمیق دارند. اول اینکه آموزش آن‌ها نسبتاً کند است و اغلب نیاز به سخت‌افزار تخصصی مانند GPGPU دارد. همچنین، مجموعه‌داده‌های بزرگ نیازمند توپولوژی‌های شبکه‌ای پیچیده هستند. با این حال، استفاده از توپولوژی‌های پیچیده برای مجموعه‌داده‌های کوچک می‌تواند منجر به بیش‌برازش شود. یکی از روش‌های منظم‌سازی که در کاهش بیش‌برازش مؤثر بوده، افزودن لایه‌ی Dropout است[60]

نوعی از VAE با نام β–VAE برای تشخیص ناهنجاری پیشنهاد شده است [61،62]. این نوع جدید، یک ابرپارامتر تازه به‌نام β معرفی می‌کند که ظرفیت رمزگذاری گلوگاه پنهان را محدود کرده و نمایش‌های پنهان را به سمت تفکیک‌پذیری سوق می‌دهد. شایان ذکر است که VAE استاندارد را می‌توان حالت خاصی از این نوع جدید در نظر گرفت، به‌طوری‌که زمانی که $\beta = 1$ باشد، این دو معادل هستند.

3.2. Feature selection

مجموعه‌داده‌های نامتوازن با ابعاد بالا یکی از چالش‌های اصلی در حوزه‌ی تحقیقاتی یادگیری ماشین هستند [63]. با استفاده از روش‌هایی که امکان انتخاب دستی یا خودکار زیرمجموعه‌ای مرتبط از مجموعه‌ای اولیه‌ی بزرگ و بالقوهٔ تکراری ویژگی‌ها را فراهم می‌کنند، می‌توان هم دقت و هم سربار محاسباتی روش‌های یادگیری ماشین را بهبود بخشید. مجموعه‌داده‌ای با ابعاد کمتر و تعادل مناسب همچنین می‌تواند بیش‌برازش را به‌طور قابل توجهی محدود کرده و تفسیر نتایج پیش‌بینی را آسان‌تر کند [64]. در مورد انتخاب ویژگی‌های بدون‌ناظر، نمی‌توان به متغیرهای هدف تکیه کرد، بلکه باید از معیارهای هدف دیگر و خوشه‌بندی برای بهبود دقت تشخیص استفاده کرد.

معیار کشیدگی (Kurtosis) را می‌توان برای هر ویژگی به‌عنوان یک روش انتخاب ویژگی بدون‌ناظر محاسبه کرد. کشیدگی ویژگی f_i ، برای i=1,…,n از m نمونه، ابتدا با نرمال‌سازی نمونه‌ها با میانگین صفر و انحراف معیار واحد محاسبه می‌شود:

$$k_j = \frac{f_{ij} - \mu_i}{\sigma_i}, \; j = 1, \ldots, m \qquad (3)$$

که در آن μ_i میانگین σ_i انحراف معیار ویژگی f_i هستند، و f_{ij} نمونه‌ی j ام از ویژگی i ام است. سپس کشیدگی (Kurtosis) ویژگی f_i به‌صورت زیر محاسبه می‌شود:

$$K_i = \frac{\sum_{j=1}^{m} k_j^4}{m} \qquad (4)$$

کشیدگی معیاری برای "غیر یکنواختی" داده‌هاست. این معیار شکل توزیع احتمال را توصیف می‌کند. مقدار بالای کشیدگی معمولاً با درجه‌ی بالای انحراف/نقاط پرت همراه است، اگرچه تفسیر آن بسته به زمینه‌ی کاربرد متغیر است. در مورد ما، سطح کشیدگی محاسبه‌شده برای هر ویژگی به ما اجازه می‌دهد تنها

4 در این روش، امتیاز نهایی با میانگین‌گیری از طول مسیر هر نقطه‌ی داده در درخت‌های مختلف ایزوله‌سازی به‌دست می‌آید. در اغلب سناریوها، الگوریتم جنگل ایزوله‌سازی بر این فرض عمل می‌کند که شناسایی یا ایزوله‌سازی نقاط پرت در زیرفضاهایی با بُعد کمتر که از تقسیم‌های تصادفی حاصل شده‌اند، محتمل‌تر است.

5 در مرحله‌ی آموزش، جنگل ایزوله‌سازی معادل بدون‌ناظر درخت‌های تصمیم‌گیری را می‌سازد. این درخت‌ها دودویی هستند و حداکثر دارای N گره‌ی برگ می‌باشند که برابر با تعداد نقاط داده در مجموعه‌ی آموزشی است. بنابراین، گره‌ی ریشه شامل تمام نقاط داده برای پردازش بوده و به‌عنوان وضعیت اولیه‌ی درخت ایزوله‌سازی T در نظر گرفته می‌شود. سپس، یک فهرست کاندیدا C شامل گره‌ی ریشه مقداردهی اولیه می‌شود. از این فهرست، گره‌ای به‌نام R به‌صورت تصادفی انتخاب می‌شود. سپس یک ویژگی تصادفی i برای تقسیم R به دو مجموعه‌ی R_1 و R_2 انتخاب می‌شود. این تقسیم با انتخاب یک مقدار تصادفی a از ویژگی i انجام می‌شود، به‌گونه‌ای که تمام نقاط داده در R_1 شرط $x_i \leq a$ و نقاط در R_2 شرط $x_i > a$ را ارضا کنند. این مقدار a به‌صورت یکنواخت و تصادفی از بین حداقل و حداکثر مقادیر ویژگی i در گره R انتخاب می‌شود. هر دو مجموعه‌ی R_1 و R_2 به‌عنوان فرزندان گره R در درخت T در نظر گرفته می‌شوند. اگر هر یک از R_iبیش از یک نقطه داشته باشد، به فهرست C افزوده می‌شود؛ در غیر این صورت، آن گره به‌عنوان برگ تعیین می‌گردد. این فرآیند تا زمانی که فهرست C خالی شود، تکرار می‌شود.

6 نتیجه‌ی آموزش، درخت دودویی نامتوازنی خواهد بود که در آن، گره‌های مربوط به نقاط پرت معمولاً در بُعد پایین‌تری نسبت به نقاط عادی ایزوله می‌شوند. این رویکرد ذاتاً تصادفی چندین بار تکرار شده و نتایج به‌صورت میانگین‌گیری ترکیب می‌شوند. پیچیدگی محاسباتی این روش برابر با $\Theta(N \log(N))$و پیچیدگی فضایی آن $O(N)$ است. درخت‌های ایزوله‌سازی خوشه‌های سلسله‌مراتبی از داده‌ها ایجاد می‌کنند، اما همان‌طور که در پاراگراف‌های پیشین اشاره شد، به‌شدت تصادفی هستند. این ویژگی در استفاده از این روش بر روی داده‌های واقعی ممکن است مشکلاتی ایجاد کند. با استفاده از زیرنمونه‌گیری می‌توان عملکرد محاسباتی را بهبود بخشید و به ایجاد تنوع کمک کرد که برای روش‌های ensemble حیاتی است. با این حال، تنوع همچنین از طریق فرآیند تصادفی ساخت درخت‌های ایزوله‌سازی حاصل می‌شود، حتی شاید بیشتر از زیرنمونه‌گیری. بنابراین، انتخاب صحیح ویژگی یا ویژگی i به‌شدت تحت تأثیر تصادفی بودن قرار دارد. در نتیجه، کاملاً ممکن است که مدل‌هایی با عملکرد ضعیف به‌دلیل انتخاب نامناسب ویژگی‌ها در مرحله‌ی آموزش ایجاد شوند.

6.1. فاکتور ناهنجاری محلی مبتنی بر خوشه‌بندی

چندین فرض بنیادین در روش‌های بدون‌ناظر وجود دارد. تکنیک‌های مبتنی بر مجاورت یک رخداد یا نقطه‌ی داده‌ی غیرعادی را زمانی تعریف می‌کنند که ناحیه‌ی اطراف آن به‌طور پراکنده از داده‌ها پر شده باشد. این مفهوم مجاورت را می‌توان به روش‌های مختلفی تعریف کرد.

در روش‌های مبتنی بر خوشه‌بندی عدم عضویت یک رخداد در یک خوشه از طریق فاصله‌ی آن از سایر خوشه‌ها، اندازه‌ی نزدیک‌ترین خوشه، یا ترکیبی از این عوامل سنجیده می‌شود تا امتیاز غیرعادی بودن محاسبه گردد. برخی روش‌ها به‌شدت بر خوشه‌بندی متکی هستند، به‌طوری‌که اساساً هر چیزی که نتوان به یک خوشه نسبت داد، به‌عنوان یک ناهنجاری در نظر گرفته می‌شود.

روش‌های مبتنی بر فاصله امتیاز ناهنجاری را بر اساس فاصله‌ی یک رخداد تا نزدیک‌ترین همسایگانش تعیین می‌کنند. در این حالت، رخدادهایی که فاصله‌ی زیادی با همسایگان نزدیک خود دارند، به‌عنوان ناهنجار شناخته می‌شوند. این روش‌ها معمولاً دارای دقت بالاتری هستند، اما این دقت بیشتر با هزینه‌ی محاسباتی بالاتری همراه است.

3.1. بر اساس تعداد رخدادهایی که در یک ناحیه‌ی محلی قرار دارند عمل می‌کنند. این روش‌های مبتنی بر چگالی تعداد برای تعریف چگالی محلی و محاسبه‌ی امتیاز ناهنجاری استفاده می‌شود. به‌طور کلی، روش‌های مبتنی بر .چگالی فضای داده را تقسیم‌بندی می‌کنند، در حالی‌که روش‌های خوشه‌بندی داده‌ها را گروه‌بندی می‌نمایند

عامل پرت محلی مبتنی بر خوشه‌بندی یا CBLOF یک الگوریتم مبتنی بر مجاورت است

که ترکیبی از عامل پرت محلی (LOF) و یک تکنیک خوشه‌بندی می‌باشد LOF .امتیاز ناهنجاری (یا پرت بودن) را بر اساس چگالی محلی تنظیم می‌کند. چگالی در این روش به‌صورت معکوس

میانگین فاصله‌ها تعریف می‌شود. این رویکرد باعث می‌شود که رخدادهایی که در نواحی محلی با چگالی بالا قرار دارند، حتی اگر از سایر رخدادهای اطراف خود جدا باشند، امتیاز ناهنجاری بالاتری دریافت کنند. دلیل اصلی این

در واقع، CBLOF امتیازی است که در آن ناهنجاری‌ها به‌عنوان ترکیبی از فاصله‌ی محلی تا خوشه‌های مجاور و اندازه‌ی خوشه‌هایی که هر رخداد به آن تعلق دارد، تعریف می‌شوند. بنابراین، رخدادهایی که در خوشه‌های کوچک قرار دارند و فاصله‌ی زیادی با خوشه‌های مجاور دارند، به‌عنوان ناهنجار شناسایی می‌شوند

چند ده کاربر از این شبکه به‌عنوان تنها راه دسترسی خود به اینترنت استفاده می‌کنند. در [71] یک صفحه‌ی وب برای پایش زنده‌ی GuifiSants وجود دارد که به‌صورت ساعتی به‌روزرسانی می‌شود. تحلیل‌های فنی مربوط به GuifiSants را می‌توان در [72] یافت.

مجموعه‌داده از نمونه‌های داده‌ای که هر ۵ دقیقه از هر گره جمع‌آوری شده‌اند، ساخته شده است. این مجموعه‌داده به‌صورت عمومی در Zenodo [19] در دسترس قرار دارد. این کار از طریق یک اتصال دائمی sshاز یک سرور مرکزی پایش به هر گره در شبکه‌ی مش انجام شده است، که از آن برای اجرای دستورات استاندارد سیستم استفاده شده است. خروجی این دستورات سپس تجزیه می‌شود تا داده‌ها استخراج شوند. مزیت این روش آن است که نیازی به نصب نرم‌افزار اضافی یا اعمال تغییرات در گره‌ها نیست. این موضوع از آن جهت مهم است که کاربران مالک گره‌های خود هستند. بنابراین، تنها با اجازه‌ی کاربران می‌توان یک کلید عمومی را برای دسترسی ssh به گره‌ها جهت پایش نصب کرد.

داده‌ها از طریق خواندن متغیرهای هسته‌ی لینوکس که از طریق سیستم فایل proc/ در دسترس هستند، به‌دست آمده‌اند. برای مثال، مسیر proc/net/dev/ برای خواندن شمارنده‌هایی مانند تعداد بایت‌ها و بسته‌های ارسال‌شده و دریافت‌شده از طریق هر واسط؛ proc/stat/ که اطلاعاتی درباره‌ی فعالیت هسته ارائه می‌دهد؛ proc/meminfo/ برای استفاده‌ی حافظه، و غیره. متغیرهای هسته‌ای به دو نوع تقسیم می‌شوند (i) :مقادیر مطلق، مانند میانگین بار پردازنده در یک دقیقه، و (ii) شمارنده‌هایی که به‌صورت یکنواخت افزایش می‌یابند، مانند تعداد بسته‌های ارسال‌شده. ما متغیرهای شمارنده‌محور را با تقسیم اختلاف بین دو نمونه‌ی متوالی بر اختلاف زمانی آن‌ها، به نرخ تبدیل کرده‌ایم.

برچسب‌های زمانی متناظر بر حسب ثانیه در نظر گرفته شده‌اند. نمونه‌هایی با نرخ منفی که هنگام راه‌اندازی مجدد یک گره یا زمانی که یک شمارنده به مقدار بیشینه‌ی خود می‌رسد و دوباره از ابتدا شروع می‌شود، رخ می‌دهند، حذف شده‌اند.

مجموعه‌داده شامل ویژگی‌های مرتبط با ترافیک و غیرترافیکی است. ویژگی‌های ترافیکی از شمارنده‌های موجود در مسیر proc/net/dev/ در سیستم فایل proc/ لینوکس استخراج شده‌اند. برای هر گره، شمارنده‌های مربوط به بایت‌ها و بسته‌های دریافتی و ارسال‌شده از طریق واسط‌های Ethernet و WiFi در نظر گرفته شده‌اند.

برای هر گره، مجموع مقادیر شمارنده‌ها برای هر دو نوع واسط (Ethernet وWiFi) ، و همچنین مجموع و تفاضل بایت‌ها و بسته‌های دریافتی و ارسال‌شده از طریق تمام واسط‌ها نیز لحاظ شده‌اند. یادآوری می‌شود که تمام ویژگی‌های شمارنده‌ای طبق توضیح قبلی به نرخ تبدیل شده‌اند. برای مثال، ویژگی‌های eth.txb.rate-24و sum.xb.rate-24 به‌ترتیب به نرخ ارسال بایت‌ها از طریق واسط‌های Ethernet و مجموع بایت‌های دریافتی و ارسال‌شده در گره ۲۴ اشاره دارند (نگاه کنید به شکل ۲).

ویژگی‌های غیرترافیکی شامل تعداد پردازه‌ها، میانگین بار پردازنده(CPU load) ، softirq، iowait، زمان اجرای پردازه‌ها در حالت‌های هسته و کاربر، تعداد تعویض‌های زمینه(context switches) ، و غیره هستند. برای اختصار، تمام ویژگی‌های جمع‌آوری‌شده در اینجا فهرست نشده‌اند. فهرست کامل ویژگی‌ها همراه با توضیح مختصر در مخزن عمومی مجموعه‌داده [19] قابل دسترسی است.

در مجموع، داده‌ها از ۶۳ گره جمع‌آوری شده‌اند و شامل ۲۳۸۷ ویژگی هستند. برای ارزیابی اثربخشی روش‌های یادگیری ماشین در تشخیص ناهنجاری‌ها، مراحل زیر انجام شده‌اند: در تاریخ ۱۴ آوریل ۲۰۲۱، یکی از دو دروازه‌ی مش (گره ۲۴) دچار خرابی شد و جایگزین گردید. به‌دلیل این خرابی، نمونه‌های مربوط به این گره بین ساعت ۰۱:۵۵ تا ۱۷:۴۰ روز ۱۴ آوریل در دسترس نبودند. برای مجموعه‌ی آزمایشی، نمونه‌های جمع‌آوری‌شده در بازه‌ی سه‌روزه‌ی وقوع خرابی (۱۳، ۱۴ و ۱۵ آوریل) استفاده شدند (۶۹۴ نمونه). برای مجموعه‌ی آموزشی، نمونه‌های جمع‌آوری‌شده در چهار هفته‌ی منتهی به مجموعه‌ی آزمایشی استفاده شدند (۷۲۳۷ نمونه). شکل ۲ ویژگی ترافیکی sum.xb.rate-24 مربوط به گره ۲۴ را در دوره‌های آموزشی و آزمایشی نشان می‌دهد.

5. آزمایش‌ها و نتایج

در این بخش، جزئیات آزمایش‌های انجام‌شده با استفاده از روش‌های بدون‌ناظر تشخیص ناهنجاری (AD)ارائه می‌شود. تمام آزمایش‌ها روی یک سرور IBM Power SC9221 با ۱۶۰ پردازنده‌ی Power9با فرکانس ۳,۷ گیگاهرتز، ۶۴۴ گیگابایت حافظه‌ی RAM ، و ۴ کارت گرافیک Nvidia V1002با حافظه‌ی ۳۲ گیگابایت GDDR5 و اتصال NVLink اجرا شده‌اند..

[1] https://www.ibm.com/downloads/cas/KQ4BOJ3N
[2] https://www.nvidia.com/en-us/data-center/v100/

ویژگی‌هایی را انتخاب کنیم که توزیع نسبتاً غیر یکنواختی دارند.

یکی از نقاط ضعف شناخته‌شده معیار کشیدگی این است که تعامل بین ویژگی‌های مختلف را در نظر نمی‌گیرد. کشیدگی چندبعدی را می‌توان با محاسبه‌ی کشیدگی بر اساس فاصله‌ی ماهالانوبیس (Mahalanobis) تمام نقاط داده تا مرکز ثقل بازنمایی مجدد داده‌های اصلی در یک زیرفضای پایین‌بعدی محاسبه کرد.

اختلاف میانگین مطلق (Mean Absolute Difference) تفاوت مطلق از مقدار میانگین را محاسبه می‌کند. امتیاز بالاتر نشان‌دهنده‌ی پتانسیل تمایز بیشتر است [65]. این معیار به‌صورت زیر تعریف می‌شود:

$$MAD_i = \frac{1}{m}\sum_{j=1}^{m} |f_{ij} - \mu_i| \quad (5)$$

3.2.1. مقادیر شِپلی

4 روش‌های ذکرشده در بالا راهکارهای مفیدی برای شناسایی ویژگی‌هایی هستند که ممکن است منجر به مدل پیش‌بینی با عملکرد بهتر شوند. یک رویکرد جدید و جالب، نه‌تنها برای انتخاب ویژگی‌هایی با تأثیر بالا بر پیش‌بینی، بلکه برای توضیح اینکه چرا در روش‌های بدون‌ناظر یک رخداد خاص به‌عنوان ناهنجار تشخیص داده شده، مقادیر شَپلی (Shapely values) هستند [66]. این مقادیر نخستین‌بار در مطالعه‌ی بازی‌های ائتلافی معرفی شدند. این روش بر پایه‌ی یک تابع ارزش (value function) تعریف می‌شود که با نماد v برای بازیکنان مجموعه S نمایش داده می‌شود. مقدار شَپلی نشان‌دهنده‌ی سهم یک ویژگی خاص در پرداخت نهایی است، که به‌صورت وزن‌دار و مجموع‌گیری‌شده بر تمام ترکیب‌های ممکن محاسبه می‌شود:

$$\phi_i(v) = \sum_{S\subseteq\{1,\dots,n\}\setminus\{i\}} \frac{|S|!\,(n-|S|-1)!}{n!}\left(v(S\cup\{i\}) - v(S)\right) \quad (6)$$

که در آن n مجموعه‌ی تمام بازیکنان را نشان می‌دهد، بنابراین مقدار شَپلی بازی v(S,n) برای توزیع سود کل v(n) میان بازیکنان، متناسب با سهم هر یک، استفاده می‌شود [33]. در زمینه‌ی یادگیری ماشین به‌طور کلی و تشخیص ناهنجاری (AD)به‌طور خاص، یک بازیکن i متناظر با یکی از ویژگی‌های مجموعه‌داده است؛ به این ترتیب، n نشان‌دهنده‌ی تعداد کل ویژگی‌های ورودی است. به‌طور معکوس، مقدار شَپلی برای ویژگی $i \in n$ ، یعنی $\varphi_i(v)$ ، میانگین وزن‌دار سهم نهایی آن ویژگی است. بر اساس معادله‌ی (6)، می‌توان پیش‌بینی را برای مقادیر ویژگی‌های موجود در مجموعه‌ی S — که زیرمجموعه‌ای از ویژگی‌های استفاده‌شده برای آموزش مدل است — محاسبه کرد، به‌طوری‌که این مقادیر بر اساس ویژگی‌هایی که در مجموعه‌ی S نیستند، به‌صورت حاشیه‌ای (marginalized) در نظر گرفته می‌شوند[67]

$$v_x(S) = \int f(x_1, \dots, x_n)\, dP(x \notin S) - \mathrm{E}_X[f(X)] \quad (7)$$

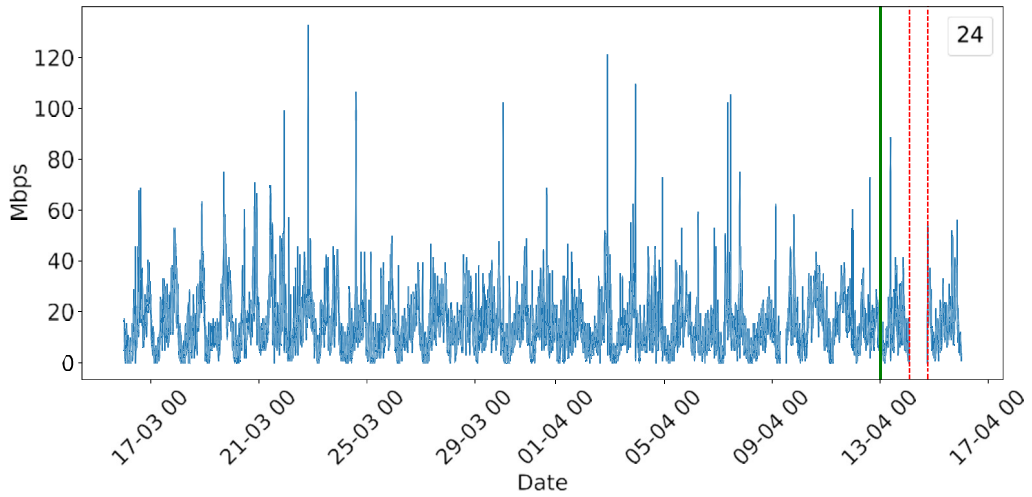
در زمینه‌ی مدل‌های یادگیری ماشین، Xبرداری از مقادیر ویژگی‌های نمونه‌ای است که قرار است توضیح داده شود، و n تعداد ویژگی‌ها را نشان می‌دهد. مقادیر شَپلی (Shapely values) دارای خاصیت تقارن هستند؛ به این معنا که اگر سهم دو ویژگی برابر باشد، مقدار شَپلی آن‌ها نیز برابر خواهد بود، و ویژگی‌هایی که سهمی در پیش‌بینی ندارند، مقدار شَپلی صفر خواهند داشت. این روش همچنین دارای خاصیت بهره‌وری (efficiency) است، به این معنا که مجموع سهم ویژگی‌ها برابر با تفاوت بین پیش‌بینی برای x و میانگین پیش‌بینی‌ها خواهد بود.

مسئله‌ی اصلی در محاسبه‌ی مقادیر شَپلی برای انتخاب ویژگی‌ها این است که این مقادیر نیازمند یک مجموعه‌داده برچسب‌خورده یا پیش‌بینی برای توضیح هستند. در مورد ما، روش‌های بدون‌ناظر مورد استفاده قرار گرفته‌اند و هدف اولیه، ارائه‌ی توضیحی برای اینکه چرا یک رخداد خاص به‌عنوان ناهنجار شناسایی شده، می‌باشد. این کاربرد به‌خوبی با مقادیر شَپلی هم‌خوانی دارد. با این حال، می‌توان اهمیت ویژگی‌ها را بر اساس رخدادهای ناهنجار شناسایی‌شده و مقادیر شَپلی محاسبه‌شده نیز تعیین کرد. این مرحله می‌تواند به‌طور قابل‌توجهی فضای ویژگی‌های مجموعه‌داده را کاهش دهد و در عین حال، پتانسیل افزایش نرخ شناسایی رخدادهای ناهنجار را نیز داشته باشد.

4. مجموعه‌داده

در این مقاله، ما بر یک شبکه‌ی بی‌سیم اجتماعی (WCN) به‌نام GuifiSants [13] تمرکز خواهیم داشت GuifiSants .فعالیت خود را در سال 2009 آغاز کرده و بخشی از شبکه‌ی Guifi.net محسوب می‌شود. گره‌های GuifiSants شامل انتن‌هایی هستند که با توزیع لینوکس [68] OpenWrt فلش شده‌اند و پروتکل مسیریابی مش BMX6 [69،70] را اجرا می‌کنند GuifiSants .یک شبکه‌ی بی‌سیم اجتماعی است که در یکی از محله‌های شهر بارسلونا (اسپانیا) راه‌اندازی شده است. به نام Sants شناخته می‌شود. در سال ۲۰۱۲، دانشگاه پلی‌تکنیک کاتالونیا یا UPC به این پروژه پیوست.

هدف از پیوستن UPC به GuifiSants ، انجام فعالیت‌های پژوهشی بود. در زمان نگارش این مقاله، حدود ۶۰ گره در شبکه‌ی GuifiSantsفعال هستند GuifiSants .یک شبکه‌ی عملیاتی محسوب می‌شود که در محیط واقعی مورد استفاده قرار می‌گیرد



/ig. 2. Plot of the sum.xb.rate-24 traffic feature (sum of received and transmitted traffic over node 24) during the training and testing set (left and right of the solid lined, respectively). Dashed lines show the gateway failure interval. Peaks and valleys show daily activity and inactivity periods. Dates in the *x*-axis are formatted as day–month hour.

ما از چهار روش یادگیری ماشین معرفی‌شده در بخش ۳ استفاده کرده‌ایم. این روش‌ها به‌دلیل تفاوت‌های بنیادین در رویکردشان به تشخیص ناهنجاری (AD) انتخاب شده‌اند. آزمایش‌ها با استفاده از زبان پایتون و کتابخانه‌های scikit-learn [73] برای PCA و pyod [74] برای Isolation Forest ، CBLOF و VAE اجرا شده‌اند.

آزمایش‌ها از مجموعه‌ای از مراحل شناخته‌شده پیروی می‌کنند. ابتدا داده‌های خام قالب‌بندی، پاک‌سازی و نرمال‌سازی می‌شوند. سپس چندین روش تشخیص مبتنی بر یادگیری ماشین اجرا می‌شوند تا بهترین پارامترهای تنظیمی (hyper-parameters) شناسایی شوند. پس از آن، روش‌های انتخاب ویژگی برای بهبود نتایج اولیه به‌کار گرفته می‌شوند. در نهایت، مجموعه‌ی دوم آزمایش‌ها با زیرمجموعه‌ی جدیدی از داده‌های اصلی اجرا شده و نتایج نهایی تحلیل می‌شوند. اسکریپت‌های پایتون مورد استفاده برای پردازش مجموعه‌داده‌ها در مخزن عمومی زیر در دسترس هستند:
https://github.com/llorenc/ml-comcom

مانند تمام وظایف یادگیری ماشین، درک و پاک‌سازی داده‌ها یکی از زمان‌برترین و حیاتی‌ترین مراحل است. در آزمایش‌های ما، ویژگی‌هایی که در بازه‌ی زمانی انتخاب‌شده دارای واریانس پایین بودند، حذف شدند. ویژگی‌های با واریانس پایین به‌صورت سراسری و با در نظر گرفتن تمام گره‌های شبکه‌ی مش شناسایی شده‌اند. داده‌های آموزشی شامل ۷۲۳۷ نمونه هستند که پس از حذف ویژگی‌های کم‌تنوع، هر نمونه دارای ۱۵۸۵ ویژگی است. مجموعه‌ی آزمایشی نیز شامل ۶۹۴ نمونه با همان فضای ویژگی‌های مجموعه‌ی آموزشی است.

نرمال‌سازی داده‌ها در صورت اعمال نادرست بر مجموعه‌داده‌های نامتوازن می‌تواند عملکرد روش‌های یادگیری ماشین را به‌شدت کاهش دهد. بسیار مهم است که اختلافات میان ویژگی‌های مجموعه‌داده به‌درستی نرمال شوند. مقادیر کم‌اهمیت و همچنین مقادیر غالب باید در یک بازه‌ی قابل‌قبول قرار گیرند [75]. ما روش min–max را انتخاب کرده‌ایم که داده‌ها را در بازه‌ی [0، 1] نرمال‌سازی می‌کند، بر اساس رابطه‌ی زیر:

$$X_n = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (8)$$

مقادیر نرمال‌شده حاصل که با نماد X_n نمایش داده می‌شوند، این مزیت را دارند که روابط میان ویژگی‌ها را حفظ کرده و در نتیجه، سوگیری را عملاً از بین می‌برند. افزون بر این، در برخی روش‌های یادگیری ماشین (مانند یادگیری عمیق و مدل‌های مبتنی بر شبکه‌های عصبی)، نرمال‌سازی صحیح داده‌ها نرخ همگرایی را کاهش می‌دهد.
در مورد PCA ، ما دریافتیم که اعمال نرمال‌سازی min–max صرفاً بر ویژگی‌های غیرترافیکی مناسب‌تر است. ویژگی‌های ترافیکی با تقسیم هر بردار ویژگی بر میانگین بیشینه‌ی مجموعه‌ی ویژگی‌های ترافیکی نرمال‌سازی شده‌اند. از آن‌جا که PCA نسبت به بازه‌های نرمال‌شده‌ی ویژگی‌ها بسیار حساس است، این روش باعث می‌شود که اهمیت بیشتری به ویژگی‌های ترافیکی نسبت به ویژگی‌های غیرترافیکی داده شود، در حالی که اهمیت نسبی گره‌هایی با مقادیر ترافیکی بالاتر نیز حفظ می‌شود.

بیشتر کتابخانه‌های مربوط به روش‌های بدون‌ناظر برای تشخیص ناهنجاری و نقاط پرت، نیاز دارند که یک عامل آلودگی (contamination factor) مورد انتظار برای آموزش مشخص شود. از آن‌جا که رخدادهای ناهنجار به‌ندرت اتفاق می‌افتند و ما می‌خواهیم از مثبت‌های کاذب جلوگیری کنیم. تا حد امکان، مقدار آلودگی (contamination) را روی مقدار $\alpha = 0.005$ تنظیم کرده‌ایم.

Table 1
Anomaly detection method hyper-parameters.

ML method	Parameters	Value
PCA	*PC number*	45
Isolation Forest	*n_estimators*	20
	max_samples	0.7
	max_features	1.0
CBLOF	*n_clusters*	8
	alpha	0.9
	beta	5
VAE	*encoder_neurons*	[128, 64, 32]
	decoder_neurons	[32, 64, 128]
	epochs	30
	dropout_rate	0.2
	activation	ReLU

آستانه‌ی تشخیص ناهنجاری‌ها بر اساس صدک α-1 از مجموعه‌ی آموزشی تعیین می‌شود. برای مجموعه‌ی آموزشی ما با ۷۲۳۷ نمونه، این عامل آلودگی منجر به $\alpha \times 7237 = 37$ ناهنجاری می‌شود. توجه داشته باشید که در مجموعه‌ی آزمایشی شامل ۶۹۴ نمونه، تحت شرایط عادی انتظار می‌رود که حدود $\alpha \times 694 = 4$ ناهنجاری وجود داشته باشد. با این حال، به‌دلیل خرابی دروازه‌ی شبکه که در بازه‌ی آزمایش رخ داده، انتظار داریم که روش‌های مختلف یادگیری ماشین تعداد بیشتری ناهنجاری را شناسایی کنند.
برای هر یک از این الگوریتم‌ها، لازم بود که پارامترهای تنظیمی (hyper-parameters) به‌صورت دستی تنظیم شوند تا رخدادهای ناهنجار در بازه‌ی زمانی مشخص‌شده که می‌دانیم ناهنجاری در آن رخ داده، گروه‌بندی شوند. جدول ۱ پارامترهای تنظیمی مورد استفاده برای هر روش تشخیص ناهنجاری را نشان می‌دهد. این مقادیر.

بهینه‌سازی شده‌اند و یک فضای تنظیم دستی برای پارامترهای تنظیمی پارامترها با استفاده از روش جستجوی تصادفی فضای جستجوی پارامترها (parameter search space) :
در مورد PCA ، تنها پارامتر قابل تنظیم، تعداد مؤلفه‌های اصلی (PC) است. برای تعیین تعداد مؤلفه‌ها، از روش رایج انتخاب تعداد ابعادی استفاده کرده‌ایم که ۹۵٪ از واریانس باقی‌مانده را حفظ می‌کند.
در مورد CBLOF ، از مقادیر پیش‌فرض برای تمام پارامترها استفاده شده است، چرا که این مقادیر بهترین نتایج را ارائه داده‌اند.
پارامترهای VAE شامل نرخ dropout نیز می‌شوند که به تعمیم‌پذیری مدل کمک کرده و از بیش‌برازش (overfitting) جلوگیری می‌کند. ساختار بخش‌های encoder و decoder در VAE نیز به‌صورت متقارن طراحی شده‌اند.

یکی از ملاحظات مهم این است که اندازه‌ی دسته (batch size) را برابر با ۳۲ تنظیم کرده‌ایم. هرچند سخت‌افزاری که آزمایش‌ها روی آن اجرا شده توانایی پردازش دسته‌های بسیار بزرگ‌تری را دارد، اما برای هم‌راستایی با سخت‌افزارهای محدودتر مانند برد Nvidia Jetson Nano[3]، از مقدار کوچک‌تری استفاده کرده‌ایم. رویکرد مشابهی نیز در تعیین تعداد دوره‌های آموزش (training epochs) اتخاذ شده است. ما آگاه هستیم که اندازه‌ی دسته و تعداد دوره‌های آموزش هر دو از پارامترهای تنظیمی مهم هستند، اما هدف نهایی ما این است که ببینیم.

[3] https://developer.nvidia.com/embedded/jetson-nano-developer-kit

سپس، آستانه‌ای برای امتیاز (score) تعیین می‌شود تا مشخص شود چه زمانی یک نمونه به‌عنوان ناهنجار در نظر گرفته می‌شود.

Table 2
Results of ML method based experiments.

ML method	Num. of features	Training [s]	Inference [s]	Testing anomalies
PCA	(all) 1585	1.25	0.14	151/122
	(only traffic) 880	0.88	0.11	13/11
Isolation Forest	(all) 1585	2.33	1.04	74/69
	(only traffic) 880	0.97	0.37	10/4
CBLOF	(all) 1585	18.28	2.71	125/115
	(only traffic) 880	3.42	0.09	3/2
VAE	(all) 1585	58.61	1.09	134/122
	(only traffic) 880	39.36	0.49	23/12

نحوه‌ی عملکرد روش‌های یادگیری ماشین ذکرشده در سناریوهایی از نوع Edge/Fog مورد بررسی قرار گرفته است. باید توجه داشت که به‌دلیل نبود مجموعه‌داده برچسب‌خورده، استفاده از تکنیک‌های معمول بهینه‌سازی پارامترهای تنظیمی عملاً غیرممکن است. در عوض، تمرکز ما بر یافتن پارامترهایی است که منجر به مدلی با پایداری و تکرارپذیری بالا شوند. در این مطالعه، این موضوع با بررسی مدل‌هایی سنجیده شده که به‌طور پیوسته ناهنجاری‌ها را در بازه‌ی آزمایشی شناسایی می‌کنند.

شایان ذکر است که پارامترهای تنظیمی الگوریتم‌ها به‌صورت تصادفی یا مستقل انتخاب نشده‌اند، چرا که در بسیاری موارد با یکدیگر مرتبط هستند. اگرچه در آزمایش‌های ما از تکنیک‌های بهینه‌سازی بدون راهنما (unguided optimization) استفاده شده، فضای پارامترهای تنظیمی برای هر الگوریتم پس از بررسی‌های قابل‌توجه تعریف شده است.

برای مثال، در موردIsolation Forest ، تعداد کم تخمین‌گرها (estimators) همراه با حداکثر تعداد ویژگی‌ها و نمونه‌ها می‌تواند منجر به بیش‌برازش مدل شود. در آزمایش‌های ما، تعداد تخمین‌گرها در بازه‌ی [10، 100]، اندازه‌ی نمونه در بازه‌ی [0.3، 1.0] و تعداد ویژگی‌ها در بازه‌ی [0.2، 10.0] تعریف شده‌اند.

به‌طور مشابه، در موردVAE ، تعداد دوره‌های آموزش (epochs) در بازه‌ی [5، 100] و نرخ dropout در بازه‌ی [0.1، 0.8] تنظیم شده‌اند. سه نوع تابع فعال‌سازی مورد آزمایش قرار گرفته‌اند'relu'، 'elu'، 'parametric_relu']ساختار encoder و decoder به‌صورت متقارن طراحی شده‌اند و معمولاً الگوی نزولی و صعودی را دنبال می‌کنند. در آزمایش‌های ما، تعداد اولیه‌ی نورون‌ها در encoder در بازه‌ی [32، 512] تعریف شده است.

می‌توان گفت که تنظیم Isolation Forest دشوارترین بخش بوده است. همان‌طور که در بخش ۳،۲ توضیح داده شده، Isolation Forestشامل مراحل متعددی است که به‌طور قابل‌توجهی تحت تأثیر تصادفی بودن قرار دارند. این ویژگی در بسیاری موارد مفید است، اما می‌تواند منجر به نتایج ناسازگار شود. اگر از بذر تصادفی (random seed) استفاده نشود، هر بار آموزش مدل ممکن است نتایج متفاوتی تولید کند. پس از صدها اجرای آزمایشی، تصمیم گرفتیم از تعداد نسبتاً کمی تخمین‌گر (۲۰ عدد)، حداکثر تعداد ویژگی‌ها برای آموزش هر تخمین‌گر، و اندازه‌ی نمونه‌ی ۰،۷ استفاده کنیم تا از بیش‌برازش جلوگیری شود.

نتایج تمام آزمایش‌ها در جدول ۲ ارائه شده‌اند. برای هر الگوریتم، زمان آموزش و استنتاج(inference) ، تعداد ناهنجاری‌های شناسایی‌شده در مجموعه‌ی آزمایشی، و تعداد نمونه‌هایی که در بازه‌ی خرابی دروازه قرار دارند (۱۲۲ نمونه در این بازه جمع‌آوری شده‌اند) گزارش شده‌اند. از آن‌جا که خرابی دروازه به‌وضوح یک وضعیت ناهنجار است، تعداد نمونه‌های شناسایی‌شده به‌عنوان ناهنجار در این بازه می‌تواند به‌عنوان معیار عملکرد الگوریتم‌ها در نظر گرفته شود.

جدول ۲ برای هر الگوریتم، تعداد ناهنجاری‌های شناسایی‌شده با استفاده از تمام ویژگی‌ها و فقط ویژگی‌های ترافیکی را نشان می‌دهد. با مقایسه‌ی این دو حالت، مشاهده می‌شود که استفاده از تمام ویژگی‌ها عملکرد الگوریتم‌ها را به‌طور قابل‌توجهی افزایش می‌دهد. برای مثال، در مورد VAE با استفاده از تمام ویژگی‌ها، ۱۳۴ ناهنجاری شناسایی شده‌اند که تمام ۱۲۲ نمونه‌ی مربوط به بازه‌ی خرابی دروازه را شامل می‌شوند. در حالی‌که با استفاده از ویژگی‌های ترافیکی تنها، فقط ۲۳ ناهنجاری شناسایی شده‌اند که تنها ۱۲ مورد از آن‌ها در بازه‌ی خرابی قرار دارند.

از نظر زمان اجرا، به‌وضوح مشخص است که استنتاج مدل زمان کمتری نسبت به آموزش نیاز دارد. در آزمایش‌های ما، زمان استنتاج فقط روی مجموعه‌ی آموزشی اندازه‌گیری شده است، چرا که این مجموعه به‌مراتب بزرگ‌تر از مجموعه‌ی آزمایشی است.

گام بعدی در آزمایش‌ها، بررسی عملکرد روش‌های یادگیری ماشین با تغییر اندازه‌ی مجموعه‌داده بود. برای این منظور، ابتدا تعداد نمونه‌های مورد استفاده در مجموعه‌ی آموزشی را تغییر دادیم. نتایج در شکل ۳ ارائه شده‌اند که تعداد ناهنجاری‌ها را نسبت به اندازه‌ی مجموعه‌ی آموزشی نشان می‌دهد.

نمونه‌ها از تعداد روزهایی که در محور افقی (abscissa) نشان داده شده‌اند، جمع‌آوری شده‌اند. نمونه‌های آموزشی از روز بلافاصله قبل از مجموعه‌ی آزمایشی گرفته شده‌اند و تا تمام روزهای چهار هفته‌ی قبل ادامه دارند. شکل ۳ عملکرد بسیار متفاوتی را میان روش‌های مختلف یادگیری ماشین نشان می‌دهد. در حالی‌که PCA، CBLOFو VAE در صورت استفاده از نمونه‌های تعداد کمی روز، تعداد زیادی ناهنجاری (و در نتیجه مثبت‌های کاذب) تولید می‌کنند، Isolation Forestتنها تعداد کمی ناهنجاری شناسایی می‌کند و در نتیجه تعداد زیادی منفی کاذب دارد.

از سوی دیگر، زمانی‌که از نمونه‌های بیش از ۱۳ روز استفاده می‌شود، تعداد ناهنجاری‌های شناسایی‌شده توسط VAE تقریباً ثابت باقی می‌ماند و تمام نمونه‌های مربوط به بازه‌ی خرابی دروازه را به‌درستی به‌عنوان ناهنجار شناسایی می‌کند. تعداد ناهنجاری‌های شناسایی‌شده توسط سایر روش‌ها حتی پس از استفاده از داده‌های بیش از ۲۰ روز نوسان دارد. بنابراین، می‌توان نتیجه گرفت که در آزمایش‌های ما، VAEقادر است با تعداد نمونه‌های کمتر، ناهنجاری‌ها را به‌طور قابل‌اعتماد شناسایی کند.

در مرحله‌ی بعد، روش‌های یادگیری ماشین را با تغییر تعداد ویژگی‌ها بررسی کردیم. ویژگی‌ها از یک تا تمام مجموعه، بر اساس میزان اهمیت مرتب شده و تعداد ناهنجاری‌ها محاسبه شد. برای تعیین اهمیت ویژگی‌ها، از نمودار مشارکت آماری Q در PCA (بخش ۳،۱) و روش‌های انتخاب ویژگی شرح‌داده‌شده در بخش ۳،۵ استفاده شده است.

برای محاسبه‌ی مقادیر شَپلی (Shapely values) از کتابخانه‌ی SHAP [76] استفاده شده که از چندین روش تقریبی برای محاسبه‌ی این مقادیر بهره می‌برد Isolation Forest .برای محاسبه‌ی مقادیر شَپلی در پیش‌بینی‌های مجموعه‌ی آزمایشی انتخاب شده، زیرا توسط کتابخانه‌ی SHAP به‌خوبی پشتیبانی می‌شود و از نظر زمان آموزش و استنتاج نسبتاً سریع است. تابع پیش‌بینی (payout) به‌صورت ۰ برای نمونه‌های عادی و ۱ برای نمونه‌های ناهنجار تعریف شده است.

شکل ۴ تعداد ناهنجاری‌های شناسایی‌شده در مجموعه‌ی آزمایشی و تعداد آن‌ها در بازه‌ی خرابی دروازه را با توجه به تعداد ویژگی‌های استفاده‌شده برای آموزش نشان می‌دهد. ناهنجاری‌های محاسبه‌شده با روش‌های یادگیری ماشین در هر ردیف، بر اساس نوع روش انتخاب ویژگی که در بالای هر ستون آمده، تقسیم شده‌اند. تعداد ویژگی‌ها بر اساس مقدار مطلق نزولی حاصل از هر روش انتخاب ویژگی مرتب شده‌اند.

این شکل نشان می‌دهد که روش‌های یادگیری ماشین بررسی‌شده، بسته به تعداد ویژگی‌های استفاده‌شده، رفتار بسیار متفاوتی دارند Isolation Forest .حساس‌ترین روش است؛ حتی با تعداد زیادی ویژگی، افزودن تنها یک ویژگی می‌تواند تعداد نقاط شناسایی‌شده به‌عنوان ناهنجار را به‌طور قابل‌توجهی تغییر دهد. در مقابل، VAEپایدارترین عملکرد را دارد. در واقع، با استفاده ازPCA ، MADو SHAP به‌عنوان روش انتخاب ویژگی، تعداد ناهنجاری‌های شناسایی‌شده توسط VAE تقریباً ثابت باقی می‌ماند، زمانی‌که تعداد ویژگی‌ها بیش از حدود یک‌چهارم کل مجموعه باشد.

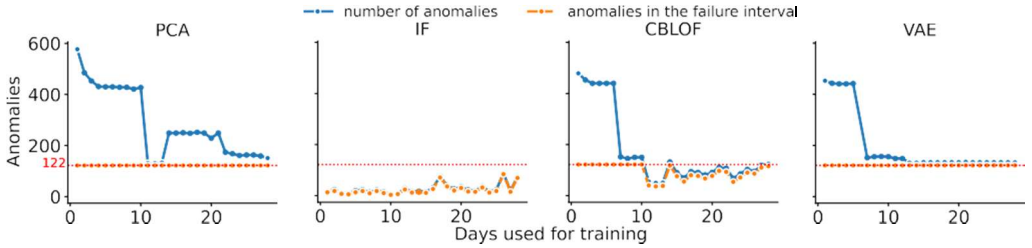
در مورد روش‌های انتخاب ویژگی، شکل ۴ نشان می‌دهد که روش kurtosis ضعیف‌ترین عملکرد را دارد. برای سایر روش‌ها، شکل ۴ نشان می‌دهد که برنده‌ی واضحی وجود ندارد و بسته به الگوریتم یادگیری ماشین، یک روش انتخاب ویژگی ممکن است نتایج بهتری ارائه دهد.

همان‌طور که پیش‌تر توضیح داده شد، پارامتر آلودگی تنظیم‌شده در مرحله‌ی آموزش منجر به شناسایی ۳۷ ناهنجاری از میان ۷۲۳۷ نمونه شد. همچنین باید اشاره کرد که بازه‌ی زمانی انتخاب‌شده به‌گونه‌ای بوده که فرض می‌شود هیچ رخداد ناهنجاری در آن رخ نداده است. با این حال، به‌جزPCA ، سایر روش‌های یادگیری ماشین عمدتاً ناهنجاری‌های مشابهی را شناسایی کرده‌اند. این موضوع به‌وضوح در شکل ۵ (a).

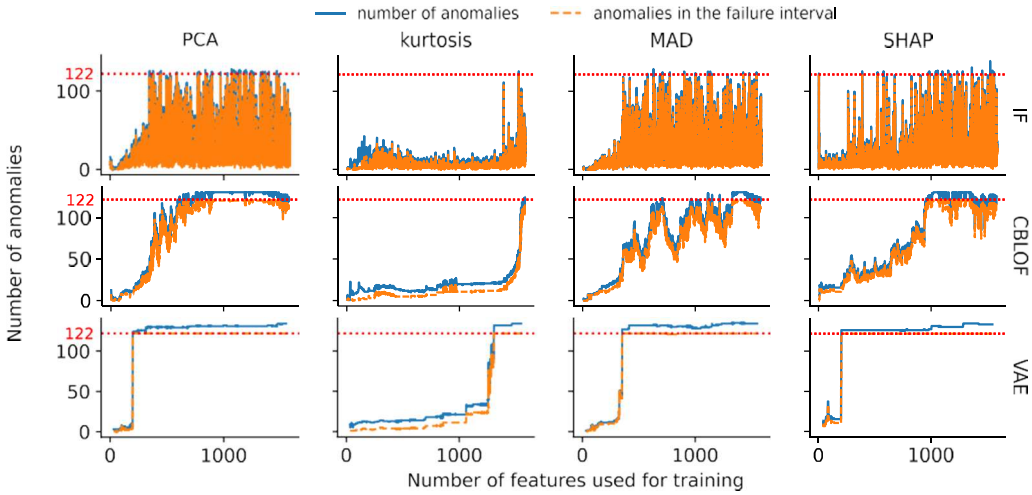
قابل مشاهده است که ناهنجاری‌های هم‌پوشان شناسایی‌شده توسط چهار روش یادگیری ماشین در مجموعه‌ی آموزشی را نشان می‌دهد. بیشترین تفاوت میان PCA و سایر روش‌ها مشاهده می‌شود که احتمالاً به‌دلیل نرمال‌سازی متفاوت اعمال‌شده در PCA است.

با این حال، اگر مجموعه‌ی آزمایشی را در نظر بگیریم، توافق بالایی میان تمام روش‌های یادگیری ماشین وجود دارد، همان‌طور که در شکل ۵ (b).نشان داده شده است. ابتدا توجه داشته باشید که تعداد ناهنجاری‌ها بسیار بیشتر از ۴ موردی است که انتظار می‌رفت. این موضوع به‌دلیل بازه‌ی خرابی دروازه است که بیشتر ناهنجاری‌ها در آن یافت می‌شوند، همان‌طور که در شکل ۵ (c).نشان داده شده است. برای مثال، شکل ۵ (c).نشان می‌دهد که PCA و VAE تمام ۱۲۲ نقطه‌ی مربوط به بازه‌ی خرابی دروازه را به‌عنوان ناهنجار شناسایی کرده‌اند. با این حال، PCAدر مجموع ۱۵۱ نقطه‌ی ناهنجار در مجموعه‌ی آزمایشی شناسایی کرده، در حالی‌که VAE تنها ۱۳۴ مورد را شناسایی کرده است. احتمالاً تعداد بیشتر نقاط شناسایی‌شده توسط PCA ناشی از بیش‌برازش مدل است.

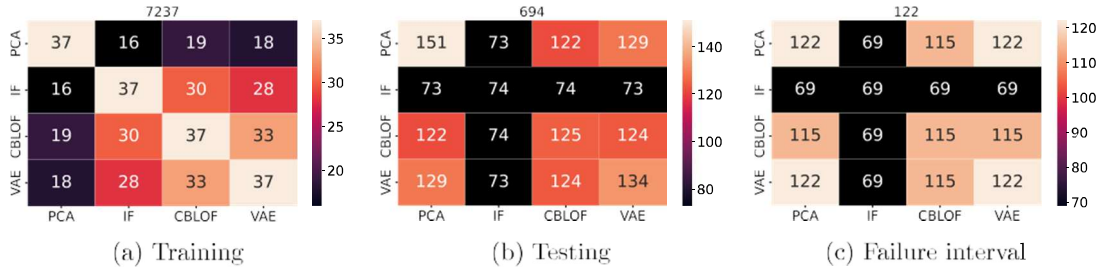
برای درک بهتر روش‌های یادگیری ماشین مقایسه‌شده، شکل ۶ یک نگاشت دوبعدی PCA از مجموعه‌ی آموزشی را نشان می‌دهد) مؤلفه‌های اصلی PC1 و (PC2 برای الگوریتم‌هایIsolation Forest ، CBLOFو VAE. توجه داشته باشید که نگاشت.



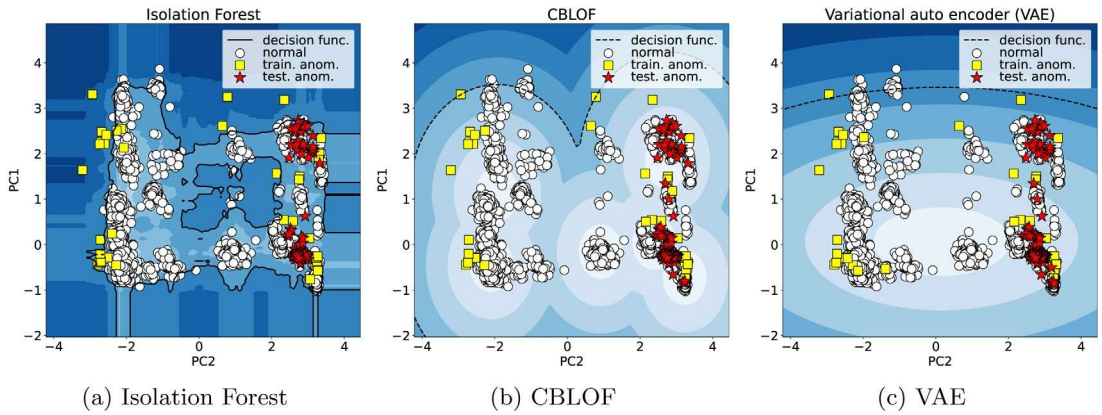
/ig. 3. Number of anomalies found in the testing set varying the number of samples used for training (samples gathered during the previous days). The figure also shows the number of anomalies obtained within the gateway failure interval (dashed line), and the number of points in this interval (122).



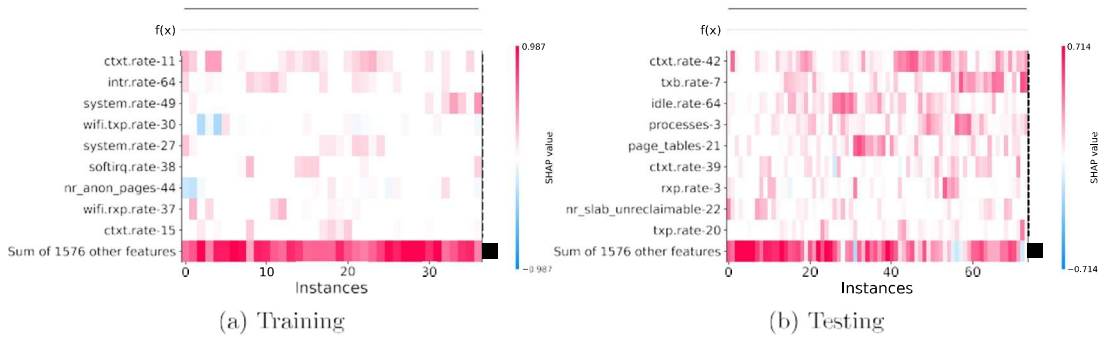
/ig. 4. Number of anomalies found in the testing set, as in Fig. 3, but varying the number of features used for training. The features are selected with different methods (shown at the top of each column).



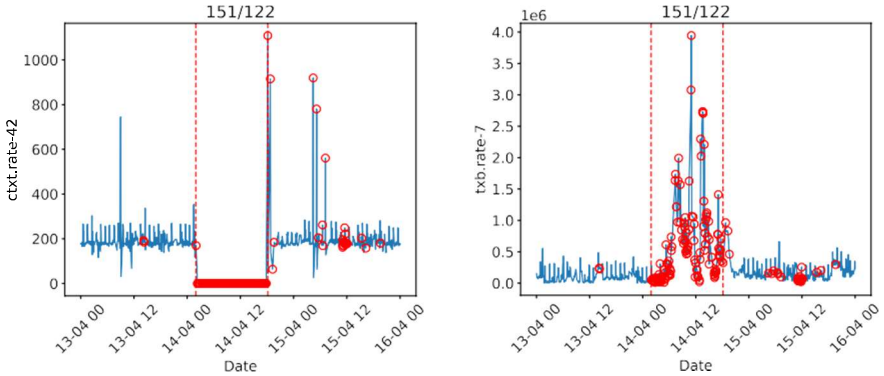
/ig. 5. Detected overlapping anomalies. Each row/column represents an AD method. Each cell of the heatmap represents the number of overlapping anomalies detected by that pair of methods. At the top of each figure there is the total number of points of the set.



/ig. 6. Decision boundaries of the ML methods re-projected using PCA with 2 components. The projected points are those in the training set (circles), anomalies due to the contamination factor in the training (squares), and anomalies in the test set (stars).



/ig. 7. SHAP heatmap plots for the training and testing anomalies obtained with Isolation Forest. The *y*-axis shows the model inputs sorted in descending order from top to bottom, for each anomalous event.



/ig. 8. Features contributing most to SHAP score during the testing set. Vertical lines correspond to the failure interval. Marked samples correspond to anomalies detected by PCA.

مقدار آلودگی تنظیم‌شده، با ویژگی‌هایی که بیشترین سهم را در مجموعه‌ی آزمایشی دارند (ناشی از خرابی دروازه)، مطابقت ندارند. این موضوع همان‌طور که انتظار می‌رفت نشان می‌دهد که ناهنجاری‌های شناسایی‌شده در دو مجموعه، ماهیت متفاوتی دارند.

شکل ۸ یک سری زمانی از دو ویژگی را نشان می‌دهد که بیشترین سهم را در تشخیص ناهنجاری‌ها در مجموعه‌ی آزمایشی دارند مطابق با شکل ۷ .(b).این دو ویژگی عبارت‌اند از نرخ تعویض زمینه‌ی پردازنده (CPU context switches) در گره ۴۲(ctxt.rate-42) ، و نرخ ارسال بایت‌ها در گره ۷ .(txb.rate-7) توجه داشته باشید که ctxt.rate-42در شرایط عادی میانگینی حدود ۲۰۰ دارد، اما در زمان خرابی دروازه مقدار آن صفر است. دلیل این موضوع آن است که در زمان خرابی، گره ۴۲ از شبکه جدا شده و هیچ نمونه‌ای از آن جمع‌آوری نشده است. این نمونه‌های مفقود به مقدار صفر تنظیم شده‌اند.

از سوی دیگر، شکل ۸ نشان می‌دهد که txb.rate-7 در زمان خرابی دروازه مقدار غیرعادی بالاتری دارد. دلیل این امر آن است که گره ۷ در نزدیکی دروازه‌ی خراب‌شده قرار دارد و در زمان خرابی، بیشتر ترافیکی که به دروازه‌ی دیگر شبکه‌ی مش هدایت شده بود را جذب کرده است.

5.1. بحث

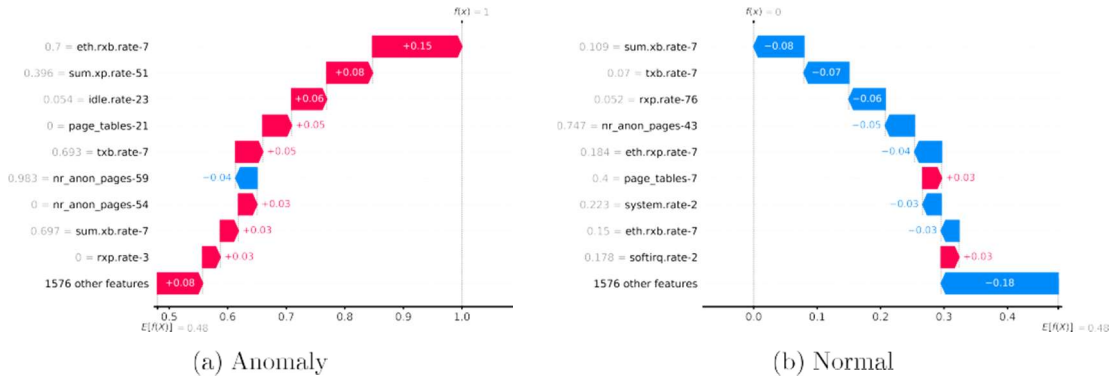
نتایج آزمایش ما جالب بوده است، ما دریافتیم که هر ۴ روش تشخیص ML انتخاب‌شده بر روی مجموعه‌داده ارائه‌شده به خوبی عمل می‌کنند[352]. متأسفانه، در مورد روش‌های انتخاب ویژگی، اینطور نیست[353]. روشی که در بیشتر موارد بهترین عملکرد را دارد، میتنی بر محاسبه مقادیر شپلی است که از نظر محاسباتی گران است[354]. ... VAE با فضای ویژگی کامل بهترین عملکرد را داشت در حالی که جنگل ایزوله‌سازی پس از انتخاب ویژگی میتنی بر مقدار شپلی، بیشترین بهبود را داشت[355]. به طور کلی، عملکرد CBLOF با زمان‌های آموزش و استنتاج نسبتاً سریع، سازگارترین بود[356]. روش انتخاب MAD نیز کاهش قابل توجهی در فضای ویژگی، به‌ویژه برای روش‌های جنگل ایزوله‌سازی و VAE، فراهم کرد

نقاط دارای مختصات یکسان هستند، زیرا از مقیاس‌بندی یکسانی برای این روش‌ها استفاده شده است. ۳۷ ناهنجاری شناسایی‌شده در مجموعه‌ی آموزشی علامت‌گذاری شده‌اند و ناهنجاری‌های شناسایی‌شده در مجموعه‌ی آزمایشی نیز نگاشت شده‌اند. علاوه بر این، روش‌ها را روی نگاشت دوبعدی مجموعه‌ی آموزشی اجرا کرده و مرز تصمیم‌گیری حاصل را ترسیم کرده‌ایم. توجه داشته باشید که ناهنجاری‌هایی که با استفاده از مجموعه‌ی آموزشی دوبعدی شناسایی می‌شوند (نقاط خارج از مرز تصمیم‌گیری)، با ناهنجاری‌هایی که با استفاده از مجموعه‌ی کامل ویژگی‌ها شناسایی می‌شوند، تفاوت قابل‌توجهی دارند. می‌توان به‌راحتی تفاوت میان نحوه‌ی «تصمیم‌گیری» هر روش یادگیری ماشین در مورد اینکه چه چیزی یک رخداد ناهنجار است و چه چیزی نیست را مشاهده کرد. مرز تصمیم‌گیری محاسبه‌شده برای Isolation Forestبه‌ویژه قابل‌توجه است. از آن‌جا که این روش بر اساس مفاهیم پایه‌ای مشابه با مدل‌های درخت تصمیم‌گیری نظارت‌شده به‌ویژه (Random Forest) بنا شده، می‌توان با اطمینان گفت که برخی ویژگی‌های خاص این مدل‌ها در Isolation Forest نیز دیده می‌شوند. شکل ۶ به‌وضوح نشان می‌دهد که بیش‌برازش مدل می‌تواند در مورد Isolation Forest یک مشکل باشد، که منجر به عملکرد ضعیف در داده‌های خارج از نمونه می‌شود.

برای تفسیر ناهنجاری‌ها، شکل ۷ یک نقشه‌ی حرارتی SHAP از ناهنجاری‌های به‌دست‌آمده از مجموعه‌های آموزشی و آزمایشی را نشان می‌دهد، در زیرشکل‌های (a) و (b) به‌ترتیب. در محور افقی (abscissa)رخدادهای ناهنجار نمایش داده شده‌اند، در حالی‌که محور عمودی (ordinate) ورودی‌های مدل را نشان می‌دهد که به‌ترتیب نزولی از بالا به پایین مرتب شده‌اند. مقادیر شَپلی محاسبه‌شده بر اساس خوشه‌بندی سلسله‌مراتبی و شباهت در توضیحات، روی یک مقیاس رنگی نمایش داده شده‌اند. در بالای هر زیرشکل، خروجی تابع f(x) با مقادیر پیش‌بینی‌شده نمایش داده شده که همواره برابر با ۱ است، زیرا تنها نمونه‌های ناهنجار ترسیم شده‌اند.

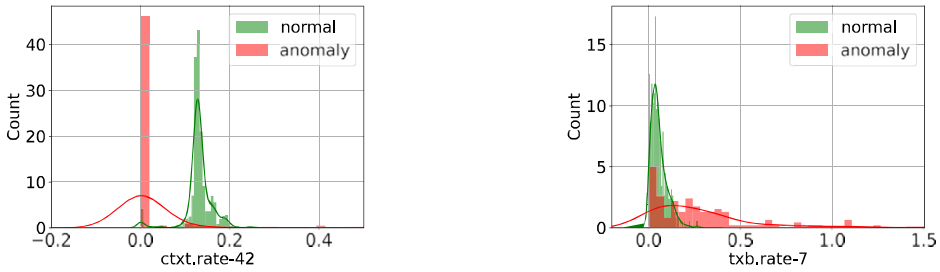
در شکل ۷ مشاهده می‌شود که شدت رنگ ناهنجاری‌های به‌دست‌آمده از مجموعه‌ی آموزشی بسیار کم‌رنگ‌تر و پراکنده‌تر است، در حالی‌که در مجموعه‌ی آزمایشی رنگ‌ها شدیدتر و پرتکرارتر هستند. این موضوع به‌وضوح نشان می‌دهد که گروهی از ویژگی‌ها به‌طور قابل‌توجهی تحت تأثیر خرابی دروازه در مجموعه‌ی آزمایشی قرار گرفته‌اند و بنابراین، برای بیشتر ناهنجاری‌های شناسایی‌شده در این مجموعه، مقدار شَپلی بالایی دریافت کرده‌اند.

همچنین جالب است که در شکل ۷ هیچ‌یک از ویژگی‌هایی که بیشترین سهم را در ناهنجاری‌های شناسایی‌شده در مجموعه‌ی آموزشی دارند



(a) Anomaly (b) Normal

/ig. 9. SHAP representation of an anomalous and normal samples randomly taken from the testing set. At the bottom is the expected value of the model. Each line shows the positive (red) or negative (blue) contribution of each feature to the prediction. The features are sorted in descending order of contribution.



(a) Feature ctxt.rate-42 (b) Feature txb.rate-7

/ig. 10. Feature separability for most impactful features. The figure shows the histograms of the scaled most impactful features for the normal and anomalous events of the testing set.

افزودن ویژگی‌ها بر اساس امتیاز MAD می‌تواند باعث افت قابل‌توجهی در عملکرد پیش‌بینی شود.

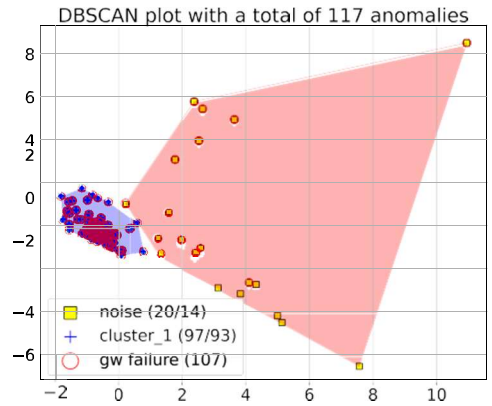
یکی از مسائل مطرح‌شده در آزمایش‌های ما این است که نمی‌دانیم چند نوع رخداد ناهنجار در مجموعه‌داده وجود دارد. هر چهار روش نمونه‌ها را به‌صورت ناهنجار یا عادی برچسب‌گذاری می‌کنند، اما هیچ‌کدام تمایزی میان انواع ناهنجاری‌ها قائل نمی‌شوند. شکل ۹ نمودارهای آبشاری (waterfall plots) حاصل از SHAP را نشان می‌دهد که یک رخداد ناهنجار و یک رخداد عادی را به‌صورت تصادفی از مجموعه‌ی آزمایشی نمایش می‌دهد؛ ویژگی‌ها به‌ترتیب نزولی مرتب شده‌اند.

در این نمودارها، پیش‌بینی‌ها به‌ترتیب $f(x)=1$ برای نمونه‌ی ناهنجار و $f(x)=0$ برای نمونه‌ی عادی هستند، با مقدار پایه‌ی $E[f(x)]=0.48$. این مقدار پایه میانگین متغیر هدف برای تمام رخدادهای موجود در مجموعه‌داده را نشان می‌دهد. ویژگی‌هایی که با رنگ قرمز نمایش داده شده‌اند، پیش‌بینی را به‌سمت ۱ (رخداد ناهنجار) سوق می‌دهند، در حالی‌که ویژگی‌های آبی آن را به‌سمت ۰ (رخداد عادی) هدایت می‌کنند. جداسازی رخدادهای پیش‌بینی‌شده به‌راحتی قابل مشاهده است. این موضوع زمانی‌که هیستوگرام‌هایی برای تأثیرگذارترین ویژگی‌های مجموعه‌ی آزمایشی ترسیم می‌شود (شکل ۱۰)، نیز صادق است.

اگرچه این اطلاعات مفید هستند، اما همچنان به این پرسش پاسخ نمی‌دهند: چند نوع ناهنجاری وجود دارد و کاهش فضای ویژگی‌ها چه تأثیری بر پیش‌بینی دارد؟ به‌احتمال زیاد، با کاهش فضای ویژگی‌ها، توانایی روش‌های تشخیص ما برای شناسایی انواع دیگر ناهنجاری‌ها نیز کاهش می‌یابد. برای مثال، اگر تنها معیارهای CPU در مجموعه‌داده موجود باشند، شناسایی ناهنجاری‌های مرتبط با حافظه تقریباً غیرممکن خواهد بود.

برای پاسخ به این پرسش، تصمیم گرفتیم ناهنجاری‌های شناسایی‌شده را با استفاده از روشی خوشه‌بندی کنیم که هر نقطه‌ی داده (رخداد) را مجبور به تعلق به یک خوشه نمی‌کند و همچنین مفهوم نویز را در نظر می‌گیرد. برای این منظور، الگوریتم HDDBSCAN (خوشه‌بندی سلسله‌مراتبی مبتنی بر چگالی با نویز) [77] انتخاب شد که خود نیز یکی از روش‌های مورد استفاده در تشخیص ناهنجاری‌هاست. این الگوریتم، DBSCAN [78]را با مقادیر مختلف پارامتر اپسیلون (\varepsilon) اجرا کرده و نتایج را یکپارچه می‌کند تا خوشه‌بندی‌ای با بیشترین پایداری نسبت به \varepsilon حاصل شود.

در DBSCAN استاندارد، پارامتر \varepsilon مشخص می‌کند که نقاط تا چه حد باید به یکدیگر نزدیک باشند تا به‌عنوان بخشی از یک خوشه در نظر گرفته شوند. تنظیم این پارامتر.



/ig. 11. Anomalous event clustering using HDDBSCAN. On the left is the anomalous event cluster (97 points), and on the right is the noise (20 points). 107 points correspond to gateway failure (circles), of which 93 are in the anomalous cluster, and 14 in the noise.

تنظیم پارامترϵ در DBSCAN یک چالش مهم است؛ اگر مقدار آن بیش از حد کوچک باشد، بخش قابل‌توجهی از داده‌ها به هیچ خوشه‌ای تخصیص نمی‌یابند و به‌عنوان نقاط پرت علامت‌گذاری می‌شوند. اگر مقدار آن بیش از حد بزرگ باشد، تمام خوشه‌ها در یک خوشه‌ی بزرگ ادغام می‌شوند HDDBSCAN . این مشکل را با ارائه‌ی خوشه‌های معنادار و نیاز بسیار کم به تنظیم پارامترها برطرف می‌کند.

در آزمایش ما، اندازه‌ی حداقل خوشه برابر با ۳۰ تنظیم شده است. برای حفظ سازگاری، از ناهنجاری‌های شناسایی‌شده توسط Isolation Forest استفاده کردیم. شکل ۱۱ نشان می‌دهد که HDDBSCAN یک خوشه شامل ۹۷ رخداد ناهنجار و ۲۰ رخداد ناهنجار به‌عنوان نویز شناسایی کرده است. از میان این ناهنجاری‌ها، ۱۰۷ مورد مربوط به بازه‌ی خرابی دروازه هستند که ۱۴ مورد از آن‌ها به‌عنوان نویز و ۹۳ مورد در خوشه قرار گرفته‌اند.

تفسیر ما از این نتایج آن است که بیشتر رخدادهای ناهنجاری شناسایی‌شده از یک نوع هستند که با خرابی دروازه مطابقت دارند.

6. از نظر پیچیدگی روش‌های یادگیری ماشین انتخاب‌شده، ابتدا باید اشاره کرد که الگوریتمی Isolation Forest (IF) است که به‌ویژه برای پردازش حجم بالای داده‌ها مناسب است و دارای پیچیدگی زمانی خطی با ضریب پایین و نیاز حافظه‌ای کم (بسته به پارامترهای تنظیمی) می‌باشد [1]. به‌طور مشابه، CBLOF نیز برای داده‌های حجیم مناسب است و در بهترین حالت دارای پیچیدگی زمانی $O(N\log N)$ است [55]. در مقابل، VAE به‌خاطر پیچیدگی محاسباتی بالا شناخته می‌شود که عمدتاً به ساختار شبکه در مرحله‌ی آموزش بستگی دارد. با این حال، در آزمایش‌های ما این پیچیدگی منجر به نتایج جالبی شد؛ VAE توانست با ساختاری ساده که مستقیماً از ساختار مجموعه‌داده ناشی می‌شود، عملکرد بهتری نسبت به سایر روش‌ها داشته باشد. همچنین، مدل‌های VAE در بهینه‌سازی پارامترهای تنظیمی پایدارترین بودند و در تمام موارد به‌جز مقادیر افراطی، مدل‌های قابل‌استفاده تولید کردند.
7. همچنین باید اشاره کرد که آزمایش‌های ما شامل چندین روش یادگیری ماشین دیگر نیز بوده‌اند. نتایجی که در این مقاله ارائه شده‌اند، بر اساس عملکرد هر مدل انتخاب شده‌اند. برای مثال، آزمایش‌هایی با روش‌های مبتنی بر GAN مانند ALAD [79] و AnoGAN [80] انجام دادیم. تفاوت اصلی این دو در آن است که ALAD بر پایه‌ی یک GAN دوطرفه ساخته شده که از محاسبات سنگین مرحله‌ی استنتاج در AnoGAN جلوگیری می‌کند. متأسفانه، نتایج تجربی در هر دو مورد ضعیف بودند . ALAD حتی با بذر تصادفی و پارامترهای ثابت، نتایج ناپایداری ارائه داد، در حالی‌که آموزش AnoGAN زمان زیادی (بیش از دو ساعت) طول کشید و نتایج محدودی داشت. شبکه‌های مبتنی بر GAN به‌خاطر مشکلاتی مانند فروپاشی مد (mode collapse) و تولید تنها بخشی از فضای داده‌ی اصلی، به‌سختی آموزش‌پذیر هستند [62].
8. به‌طور مشابه، چندین آزمایش با β-VAE انجام دادیم و مقدار β را در بازه‌ی [0.1 تا 10.0] تنظیم کردیم. با این حال، بهترین نتایج با مقدار پیش‌فرض $\beta = 1.0$ که در مقاله‌ی اصلی ارائه شده بود، به‌دست آمد. این موضوع چندان غیرمنتظره نبود، چرا که در بازتولید نتایج β-VAE نیز مشکلاتی گزارش شده‌اند [81]. در نهایت، از پیاده‌سازی Deep-SVDD [82] نیز استفاده کردیم که نتایج آن به‌طور قابل‌توجهی ضعیف‌تر از الگوریتم‌های ارائه‌شده در بخش ۴ بود.
9. برای اختصار، تمام الگوریتم‌های آزمایش‌شده را در مقاله نیاورده‌ایم و تنها سه مورد با بهترین عملکرد و اصول زیرساختی متفاوت را انتخاب کرده‌ایم. با این حال، کدها و تنظیمات پارامترهای مورد استفاده در مخزن ما در دسترس هستند.
10. ما نتیجه می‌گیریم که ضعف عملکرد پیش‌بینی در برخی روش‌های استفاده‌شده، ناشی از فضای ویژگی بسیار بزرگ مجموعه‌داده است. اگرچه استفاده از تکنیک‌های انتخاب ویژگی برای سه مدل برتر عملکرد خوبی داشته است (نگاه کنید به شکل ۳)، اما کارهای آینده بر روش‌های اضافی تمرکز خواهند داشت. تجزیه‌ی ماتریس غیرمنفی (NMF) [83] امیدبخش است، اما آزمایش‌های اولیه‌ی ما با آن نتایج محدودی داشته‌اند. در آینده، بازنگری در ساختار مجموعه‌های آموزشی و آزمایشی، شامل گراف‌های جدول‌های مسیریابی در شبکه‌ی مش و افزودن روش‌های AD جدید، در دستور کار خواهد بود.

11. نتیجه‌گیری

در دهه‌ی گذشته، روش‌های یادگیری ماشین (ML) رشد چشمگیری داشته‌اند و در حوزه‌های متعددی به‌کار گرفته شده‌اند. شبکه‌های کامپیوتری نیز از این قاعده مستثنی نیستند و مطالعات متعددی از ML برای شناسایی نفوذ در شبکه استفاده کرده‌اند. تشخیص خطا در شبکه‌های کامپیوتری یکی از کاربردهای جذاب ML است که تاکنون توجه کمی به آن شده، عمدتاً به‌دلیل نبود مجموعه‌داده‌های در دسترس. در این مقاله، تلاش کردیم این خلأ را با اجرای تشخیص ناهنجاری برای شناسایی خطا با استفاده از ML پر کنیم. روش‌های ML انتخاب‌شده در این مطالعه بر پایه‌ی چهار اصل کاملاً متفاوت هستند. نخست، تحلیل مؤلفه‌های اصلی (PCA) است؛ روشی شناخته‌شده که در صنعت تولید کاربرد گسترده‌ای دارد و بر پایه‌ی نگاشت داده‌ها به زیرفضایی با ابعاد کاهش‌یافته است که بیشترین واریانس را حفظ می‌کند. دوم، جنگل ایزوله Isolation Forest) یا (IF است که بر اساس ساخت درخت‌های تصمیم‌گیری برای ایزوله‌سازی نمونه‌ها عمل می‌کند و نمونه‌هایی را که در تعداد گام‌های کمتری ایزوله می‌شوند، به‌عنوان ناهنجار در نظر می‌گیرد. سوم، روش مبتنی بر خوشه‌بندی روش Local Outlier Factor و CBLOF بر پایه‌ی اصل خوشه‌بندی عمل می‌کنند؛ به‌طوری‌که نمونه‌هایی که فاصله‌ی بیشتری از خوشه‌های شکل‌گرفته دارند، به‌عنوان ناهنجار در نظر گرفته می‌شوند. در نهایت، از AutoEncoder های تغییرپذیر (VAE) استفاده کرده‌ایم که بر پایه‌ی مدل‌های شبکه‌ی عصبی عمیق بنا شده‌اند. تمام این روش‌ها در مطالعات پیشین به‌عنوان گزینه‌هایی مناسب برای تشخیص ناهنجاری (AD) معرفی شده‌اند.

برخلاف سایر مطالعات موجود در ادبیات، ما مجموعه‌داده‌ای از یک شبکه‌ی عملیاتی واقعی ساخته‌ایم. در این مجموعه‌داده، بازه‌ای را انتخاب کرده‌ایم که در آن یک خرابی ناگهانی در دروازه‌ی شبکه رخ داده است. این رخداد نادر به ما اجازه داد تا عملکرد روش‌های یادگیری ماشین مورد بررسی را با سنجش تعداد نمونه‌هایی که در بازه‌ی خرابی به‌عنوان ناهنجار علامت‌گذاری شده‌اند، تحلیل کنیم. نمونه‌ها هر ۵ دقیقه جمع‌آوری شده‌اند و شامل ویژگی‌های ترافیکی و غیرترافیکی مانند مصرف CPU ، مصرف حافظه و غیره هستند. دوره‌ی آموزشی شامل ۴ هفته و دوره‌ی آزمایشی شامل ۳ روز است. خرابی دروازه به‌مدت ۱۶ ساعت در میانه‌ی دوره‌ی آزمایشی رخ داده است.

نتایج اصلی به‌صورت زیر خلاصه می‌شوند:

- در صورت تنظیم مناسب، ناهنجاری‌های ناشی از خرابی دروازه به‌خوبی توسط تمام روش‌های یادگیری ماشین شناسایی می‌شوند.
- خارج از بازه‌ی خرابی، چندین جهش ترافیکی نیز به‌عنوان ناهنجاری علامت‌گذاری شده‌اند. این جهش‌ها را می‌توان نویز و حاصل الگوهای نامنظم ترافیک کاربران دانست.
- روش یادگیری عمیق مبتنی بر VAE عملکرد بهتری نسبت به سایر روش‌ها دارد، هرچند با هزینه‌ی محاسباتی بالاتر همراه است.
- در نظر گرفتن ویژگی‌های مرتبط با CPU و حافظه در کنار ویژگی‌های ترافیکی، تعداد نقاط ناهنجار شناسایی‌شده در بازه‌ی خرابی دروازه را به‌طور قابل‌توجهی افزایش می‌دهد.
- از میان روش‌های انتخاب ویژگی آزمایش‌شده، میانگین تفاوت مطلق (MAD) و مقدار شَپلی (SHAP) بهترین عملکرد را داشتند MAD .محاسبه‌ی ساده‌تری دارد، در حالی‌که SHAP درک بهتری از دلایل شناسایی ناهنجاری‌ها ارائه

CRediT authorship contribution statement

Llorenç Cerdà-Alabern: Conceptualization, Investigation, Software, Writing, Dataset gathering. **Gabriel Iuhasz:** Conceptualization, Investigation, Software, Writing. **Gabriele Gemmi:** Investigation, Writing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The dataset is available at the link to Zenodo provided in the paper.

Acknowledgments

This work has received funding through the DiPET CHIST-ERA under grant agreement PCI2019-111850-2; Spanish grant PID2019-106774RB-C21; Romanian DIPET (62652/15.11.2019) project funded via PN 124/2020; and has been partially supported by the EU research project SERRANO (101017168) and hardware resources courtesy of the Romanian Ministry of Research and Innovation UEFISCDI COCO research project PN III-P4-ID-PCE-2020-0407.

References

[1] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM Comput. Surv. 41 (3) (2009) 1–58.

[2] C.C. Aggarwal, Outlier Analysis, Springer, 2017.

[3] M. Ahmed, A.N. Mahmood, J. Hu, A survey of network anomaly detection techniques, J. Netw. Comput. Appl. (60) (2016) 19–31.

[4] D.P. Kumar, T. Amgoth, C.S.R. Annavarapu, Machine learning algorithms for wireless sensor networks: A survey, Inf. Fusion 49 (2019) 1–25.

[5] M. Mohri, A. Rostamizadeh, A. Talwalkar, Foundations of Machine Learning, MIT Press, 2018.

[6] K.P. Murphy, Probabilistic Machine Learning: An Introduction, MIT Press, 2022.

[7] L.N. Tidjon, M. Frappier, A. Mammar, Intrusion detection systems: A cross-domain overview, IEEE Commun. Surv. Tutor. 21 (4) (2019) 3639–3681.

[8] G. Fernandes, J.J. Rodrigues, L.F. Carvalho, J.F. Al-Muhtadi, M.L. Proença, A comprehensive survey on network anomaly detection, Telecommun. Syst. 70 (3) (2019) 447–489.

[9] D. Vega, R. Baig, L. Cerdà-Alabern, E. Medina, R. Meseguer, L. Navarro, A technological overview of the guifi.net community network, Comput. Netw. 9 (2) (2015) 260–278, http://dx.doi.org/10.1016/j.comnet.2015.09.023.

[10] Y. Ben David, Connecting the Last Billion (Ph.D. thesis), UC Berkeley, 2015.

[11] Guifi.net, Open, free and neutral network internet for everybody, 2021, http://guifi.net/en. (Accessed 13 January 2021).

[12] L. Cerdà-Alabern, R. Baig, L. Navarro, On the guifi.net community network economics, Comput. Netw. 168 (2020) 107067.

[13] GuifiSants, Xarxa oberta, lliure i neutral del barri de sants, 2021, http://sants.guifi.net/. (Accessed January 2021).

[14] J. Camacho, A. Pérez-Villegas, P. Garcí a Teodoro, G. Maciá-Fernández, PCA-based multivariate statistical network monitoring for anomaly detection, Comput. Secur. 59 (2016) 118–137.

[15] D.H. Hoang, H.D. Nguyen, A PCA-based method for IoT network traffic anomaly detection, in: 2018 20th International Conference on Advanced Communication Technology, ICACT, IEEE, 2018, pp. 381–386.

[16] I.K. Savvas, A.V. Chernov, M.A. Butakova, C. Chaikalis, Increasing the quality and performance of N-dimensional point anomaly detection in traffic using PCA and DBSCAN, in: 2018 26th Telecommunications Forum, TELFOR, IEEE, 2018, pp. 1–4.

[17] Munin networked resource monitoring tool, http://munin-monitoring.org.

[18] Nagios, The Industry Standard In IT Infrastructure Monitoring, https://www.nagios.org.

[19] L. Cerdà-Alabern, Dataset for Anomaly Detection in a Production Wireless Mesh Community Network, Zenodo, 2022, http://dx.doi.org/10.5281/zenodo.6169917.

[20] V. Hodge, J. Austin, A survey of outlier detection methodologies, Artif. Intell. Rev. 22 (2) (2004) 85–126.

[21] S. Northcutt, J. Novak, Network Intrusion Detection, Sams Publishing, 2002.

[22] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E.D. Kolaczyk, N. Taft, Structural analysis of network traffic flows, in: Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems, 2004 pp. 61–72.

[23] Z.R. Zaidi, S. Hakami, T. Moors, B. Landfeldt, Detection and identification of anomalies in wireless mesh networks using principal component analysis (PCA), J. Interconnect. Netw. 10 (04) (2009) 517–534.

[24] Z.R. Zaidi, S. Hakami, B. Landfeldt, T. Moors, Real-time detection of traffic anomalies in wireless mesh networks, Wirel. Netw. 16 (6) (2010) 1675–1689.

[25] C. Pascoal, M.R. De Oliveira, R. Valadas, P. Filzmoser, P. Salvador, A. Pacheco, Robust feature selection and robust PCA for internet traffic anomaly detection, in: 2012 Proceedings IEEE Infocom, IEEE, 2012, pp. 1755–1763.

[26] H. Ringberg, A. Soule, J. Rexford, C. Diot, Sensitivity of PCA for traffic anomaly detection, in: Proceedings of the 2007 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, 2007 pp. 109–120.

[27] N.L.D. Khoa, T. Babaie, S. Chawla, Z. Zaidi, Network anomaly detection using a commute distance based approach, in: 2010 IEEE International Conference on Data Mining Workshops, IEEE, 2010, pp. 943–950.

[28] A.B. Nassif, M.A. Talib, Q. Nasir, F.M. Dakalbab, Machine learning for anomaly detection: A systematic review, IEEE Access 9 (2021) 78658–78700, http://dx.doi.org/10.1109/ACCESS.2021.3083060.

[29] S. Goldstein, A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data, PLoS One 11 (4) (2016) 1–31, http://dx.doi.org/10.1371/journal.pone.0152173.

[30] M. Ahmed, A.N. Mahmood, Novel approach for network traffic pattern analysis using clustering-based collective anomaly detection, Ann. Data Sci. 2 (2015) 111–130.

[31] X. Chun-Hui, S. Chen, B. Cong-Xiao, L. Xing, Anomaly detection in network management system based on isolation forest, in: 2018 4th Annual International Conference on Network and Information Systems for Computers, ICNISC, 2018, pp. 56–60, http://dx.doi.org/10.1109/ICNISC.2018.00019.

[32] G. Pang, C. Shen, L. Cao, A.V.D. Hengel, Deep learning for anomaly detection: A review, ACM Comput. Surv. 54 (2) (2021) http://dx.doi.org/10.1145/3439950.

[33] N. Takeishi, Y. Kawahara, On anomaly interpretation via Shapley values, 2020, arXiv:2004.04464.

[34] L. Antwarg, R.M. Miller, B. Shapira, L. Rokach, Explaining anomalies detected by autoencoders using Shapley additive explanations, Expert Syst. Appl. 186 (2021) 115736, http://dx.doi.org/10.1016/j.eswa.2021.115736.

[35] C. Zhou, R.C. Paffenroth, Anomaly detection with robust deep autoencoders, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '17, Association for Computing Machinery, New York, NY, USA, 2017, pp. 665–674, http://dx.doi.org/10.1145/3097983.3098052.

[36] D.P. Kingma, M. Welling, Auto-encoding variational Bayes, 2014, arXiv:1312.6114.

[37] M. Moulay, R.G. Leiva, V. Mancuso, P.J.R. Maroni, A.F. Anta, Ttrees: Automated classification of causes of network anomalies with little data, in: 2021 IEEE 22nd International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM, IEEE, 2021, pp. 199–208.

[38] K. Sequeira, M. Zaki, ADMIT: Anomaly-based data mining for intrusions, in: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '02, Association for Computing Machinery, New York, NY, USA, 2002, pp. 386–395, http://dx.doi.org/10.1145/775047.775103.

[39] Y.F. Zhang, Z.Y. Xiong, X.Q. Wang, Distributed intrusion detection based on clustering, in: 2005 International Conference on Machine Learning and Cybernetics, vol. 4, 2005, pp. 2379–2383 Vol. 4, http://dx.doi.org/10.1109/ICMLC.2005.1527342.

[40] M.H. Bhuyan, D.K. Bhattacharyya, J.K. Kalita, An effective unsupervised network anomaly detection method, in: Proceedings of the International Conference on Advances in Computing, Communications and Informatics, ICACCI '12, Association for Computing Machinery, New York, NY, USA, 2012, pp. 533–539, http://dx.doi.org/10.1145/2345396.2345484.

[41] N. Hu, Z. Tian, H. Lu, X. Du, M. Guizani, A multiple-kernel clustering based intrusion detection scheme for 5G and IoT networks, Int. J. Mach. Learn. Cybern. 12 (2021) http://dx.doi.org/10.1007/s13042-020-01253-w.

[42] M.E. Otey, A. Ghoting, S. Parthasarathy, Fast distributed outlier detection in mixed-attribute data sets, Data Min. Knowl. Discov. 12 (2–3) (2006) 203–228, http://dx.doi.org/10.1007/s10618-005-0014-6.

[43] M. Bhuyan, D.K. Bhattacharyya, J. Kalita, A multi-step outlier-based anomaly detection approach to network-wide traffic, Inform. Sci. 348 (2016) http://dx.doi.org/10.1016/j.ins.2016.02.023.

[44] P. Casas, J. Mazel, P. Owezarski, Unsupervised network intrusion detection systems: Detecting the unknown without knowledge, Comput. Commun. 35 (7) (2012) 772–783, http://dx.doi.org/10.1016/j.comcom.2012.01.016.

[45] O. Iraqi, H. El Bakkali, Application-level unsupervised outlier-based intrusion detection and prevention, Secur. Commun. Netw. 2019 (2019) 1–13, http://dx.doi.org/10.1155/2019/8368473.

[46] M. Khan, Rule based network intrusion detection using genetic algorithm, Int. J. Comput. Appl. 18 (2011) 26–29, http://dx.doi.org/10.5120/2303-2914.

[47] H. Alsaadi, R. Almuttairi, O. Ucan, O. Bayat, An adapting soft computing model for intrusion detection system, Comput. Intell. 01 (2021) http://dx.doi.org/10.1111/coin.12433.

[48] A. Shenfield, D. Day, A. Ayesh, Intelligent intrusion detection systems using artificial neural networks, ICT Express 4 (2) (2018) 95–99, http://dx.doi.org/10.1016/j.icte.2018.04.003, SI on Artificial Intelligence and Machine Learning.

[49] C.F. Alcala, S.J. Qin, Analysis and generalization of fault diagnosis methods for process monitoring, J. Process Control 21 (3) (2011) 322–330.

[50] P. Miller, R.E. Swanson, C.E. Heckler, Contribution plots: A missing link in multivariate quality control, Appl. Math. Comput. Sci. 8 (4) (1998) 775–792.

[51] C.C. Aggarwal, S. Sathe, Theoretical foundations and algorithms for outlier ensembles, SIGKDD Explor. Newsl. 17 (1) (2015) 24–47, http://dx.doi.org/10.1145/2830544.2830549.

[52] C.C. Aggarwal, Outlier ensembles: Position paper, SIGKDD Explor. Newsl. 14 (2) (2013) 49–58, http://dx.doi.org/10.1145/2481244.2481252.

[53] F.T. Liu, K.M. Ting, Z.H. Zhou, Isolation-based anomaly detection, ACM Trans. Knowl. Discov. Data 6 (1) (2012) http://dx.doi.org/10.1145/2133360.2133363.

[54] F.T. Liu, K.M. Ting, Z.H. Zhou, Isolation forest, in: 2008 Eighth IEEE International Conference on Data Mining, 2008, pp. 413–422, http://dx.doi.org/10.1109/ICDM.2008.17.

[55] Z. He, X. Xu, S. Deng, Discovering cluster-based local outliers, Pattern Recognit. Lett. 24 (9) (2003) 1641–1650, http://dx.doi.org/10.1016/S0167-8655(03)00003-5.

[56] M.M. Breunig, H.P. Kriegel, R.T. Ng, J. Sander, LOF: Identifying density-based local outliers, SIGMOD Rec. 29 (2) (2000) 93–104, http://dx.doi.org/10.1145/335191.335388.

[57] R. Zheng, J. Gu, Anomaly detection for power system forecasting under data corruption based on variational auto-encoder, in: 8th Renewable Power Generation Conference, RPG 2019, 2019, pp. 1–6, http://dx.doi.org/10.1049/cp.2019.0461.

[58] R. Yao, C. Liu, L. Zhang, P. Peng, Unsupervised anomaly detection using variational auto-encoder based feature extraction, in: 2019 IEEE International Conference on Prognostics and Health Management, ICPHM, 2019, pp. 1–7, http://dx.doi.org/10.1109/ICPHM.2019.8819434.

[59] Y. Aizenbud, O. Lindenbaum, Y. Kluger, Probabilistic robust autoencoders for anomaly detection, 2021, arXiv:2110.00494.

[60] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (56) (2014) 1929–1958, http://jmlr.org/papers/v15/srivastava14a.html.

[61] C.P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, A. Lerchner, Understanding disentangling in β-VAE, CoRR (2018) arXiv:1804.03599.

[62] L. Zhou, W. Deng, X. Wu, Unsupervised anomaly localization using VAE and beta-VAE, 2020, http://dx.doi.org/10.48550/ARXIV.2005.10686.

[63] J. Cai, J. Luo, S. Wang, S. Yang, Feature selection in machine learning: A new perspective, Neurocomputing 300 (2018) 70–79, http://dx.doi.org/10.1016/j.neucom.2017.11.077.

[64] C. Suman, S. Tripathy, S. Saha, Building an effective intrusion detection system using unsupervised feature selection in multi-objective optimization framework, 2019, arXiv preprint arXiv:1905.06562.

[65] A.J. Ferreira, M.A.T. Figueiredo, Efficient feature selection filters for high-dimensional data, Pattern Recognit. Lett. 33 (13) (2012) 1794–1804, http://dx.doi.org/10.1016/j.patrec.2012.05.019.

[66] L.S. Shapley, 17. A value for n-person games, in: H.W. Kuhn, A.W. Tucker (Eds.), Contributions to the Theory of Games (AM-28), vol. II, Princeton University Press, 2016, pp. 307–318, http://dx.doi.org/10.1515/9781400881970-018.

[67] C. Molnar, Interpretable Machine Learning, Independently published, 2022.

[68] OpenWrt Project, OpenWrt project: Welcome to the OpenWrt project, 2021, https://openwrt.org/. (Accessed January 2021).

[69] BMX6 mesh networking protocol, http://bmx6.net. (Accessed January 2021).

[70] L. Cerdà-Alabern, A. Neumann, L. Maccari, Experimental evaluation of BMX6 routing metrics in a 802.11an wireless-community mesh network, in: 2015 3rd International Conference on Future Internet of Things and Cloud, 2015 pp. 770–775.

[71] GuifiSants, qMp Sants-UPC, 2021, http://dsg.ac.upc.edu/qmpsu. (Accessed January 2021).

[72] L. Cerdà-Alabern, A. Neumann, P. Escrich, Experimental evaluation of a wireless community mesh network, in: The 16th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, MSWiM'13, ACM, Barcelona, Spain, 2013.

[73] F. Pedregosa, et al., Scikit-learn: Machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.

[74] Y. Zhao, Z. Nasrullah, Z. Li, Pyod: A Python toolbox for scalable outlier detection, J. Mach. Learn. Res. 20 (96) (2019) 1–7, http://jmlr.org/papers/v20/19-011.html.

[75] A. Mahfouz, A. Abuhussein, D. Venugopal, S. Shiva, Ensemble classifiers for network intrusion detection using a novel network attack dataset, Future Internet 12 (11) (2020) http://dx.doi.org/10.3390/fi12110180.

[76] S.M. Lundberg, S.I. Lee, A unified approach to interpreting model predictions, in: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, pp. 4765–4774.

[77] L. McInnes, J. Healy, S. Astels, Hdbscan: Hierarchical density based clustering, J. Open Source Softw. 2 (11) (2017) 205, http://dx.doi.org/10.21105/joss.00205.

[78] M. Ester, H.P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD '96, AAAI Press, 1996, pp. 226–231.

[79] H. Zenati, M. Romain, C.S. Foo, B. Lecouat, V.R. Chandrasekhar, Adversarially learned anomaly detection, 2018, http://dx.doi.org/10.48550/ARXIV.1812.02288.

[80] T. Schlegl, P. Seeböck, S.M. Waldstein, U. Schmidt-Erfurth, G. Langs, Unsupervised anomaly detection with generative adversarial networks to guide marker discovery, 2017, http://dx.doi.org/10.48550/ARXIV.1703.05921.

[81] M. Fil, M. Mesinovic, M. Morris, J. Wildberger, Beta-VAE reproducibility: Challenges and extensions, 2021, http://dx.doi.org/10.48550/ARXIV.2112.14278, https://arxiv.org/abs/2112.14278.

[82] P. Liznerski, L. Ruff, R.A. Vandermeulen, B.J. Franks, M. Kloft, K.R. Müller, Explainable deep one-class classification, 2020, http://dx.doi.org/10.48550/ARXIV.2007.01760.

[83] H. Alshammari, O. Ghorbel, M. Aseeri, M. Abid, Non-negative matrix factorization (NMF) for outlier detection in wireless sensor networks, in: 2018 14th International Wireless Communications & Mobile Computing Conference, IWCMC, 2018, pp. 506–511, http://dx.doi.org/10.1109/IWCMC.2018.8450421.