VOL. 39 Nº2 2022
https://doi.org/10.22491/IMECTA.10831.pt

Detecção de Ataques Cibernéticos utilizando Aprendizado de Máquina: uma revisão

Ricardo da Silveira Lopes*, Julio Cesar Duarte e Ronaldo Ribeiro Goldschmidt
Instituto Militar de Engenharia (IME)
Praça Gen. Tibúrcio, 80 – Urca, Rio de Janeiro – RJ, 22290-270
***ricardo.lopes@ime.eb.br**

RESUMO: Com a disponibilização pública de bases de dados simuladas de detecção de intrusão, o Aprendizado de Máquina vem sendo empregado, cada vez mais, em estudos de detecção de ataques cibernéticos. Se, por um lado, tem-se destacado o desempenho (precisão e abrangência) obtido, por outro, tem havido uma carência na análise crítica sobre o que de fato foi aprendido pelo modelo, visando concluir se haverá ou não a manutenção desse desempenho em aplicações reais. Nesse sentido, técnicas de explicabilidade surgem como uma possibilidade promissora na execução dessa tarefa, uma vez que, usualmente, vem sendo negligenciada a análise da Taxa de Falso Positivo desses modelos, o que pode se tornar um problema importante, com o aumento da velocidade e quantidade de dados trafegados pela internet. Esta pesquisa se propõe a levantar discussões sobre esses problemas, apresentando alguns artigos a eles relacionados.

PALAVRAS-CHAVE: Detecção de Intrusão. Aprendizado de Máquina. Explicabilidade. Taxa de Falso Positivo.

ABSTRACT: With the public availability of simulated intrusion detection datasets, Machine Learning has been increasingly used in cyber attack detection work. Despite the fact that the performance (precision and recall) has been highlighted, on the other hand, there has been a lack of critical analysis of what was actually learned by the model, with the intention to conclude whether or not this performance will be maintained in real applications. In this sense, explainability techniques appear as a promising possibility in the execution of this task, since the analysis of the False Positive Rate of these models has usually been neglected. This can become an important problem, with the increase in speed and amount of data transmitted over the internet. This research proposes to raise discussions about these problems, presenting some articles related to them.

KEYWORDS: Intrusion Detection. Machine Learning. Explainability. XAI. False Positive Rate.

1. Introdução

O surgimento da internet revolucionou a vida moderna, possibilitando a realização de inúmeras atividades de forma online. Compras, transações bancárias, reuniões, aulas, cursos, interações via redes sociais, comunicação por texto, voz e vídeo, gerenciamento remoto etc., tornaram-se atividades cotidianas na sociedade [1]. Por outro lado, o aumento da utilização da informática e de ferramentas online trouxe consigo vulnerabilidades amplamente exploradas por indivíduos e organizações mal-intencionadas por meio de ações maliciosas no ciberespaço. Essas ações são conhecidas por ataques cibernéticos ou intrusões e causam considerável prejuízo a usuários e empresas que inevitavelmente utilizam a Internet. Normalmente, esses ataques possuem finalidades diversas, como, por exemplo, obter benefícios financeiros de forma ilícita, prejudicar instituições, propagar ideologias e até mesmo terrorismo [2].

Dessa forma, o ataque cibernético tem sido um problema cada vez maior conforme a sociedade vai se tornando mais dependente da tecnologia da informação. Dois fatores são responsáveis por esse problema: a existência de vulnerabilidades nos sistemas de informação e a presença de agentes com potencial para explorar tais vulnerabilidades. Esses agentes podem ser indivíduos, grupos e até mesmo nações.

Além disso, é inevitável a crescente inserção de computadores e redes de dados nos processos de gerenciamento, monitoração, automação e controle de infraestruturas críticas, tais como usinas de geração de energia, sistemas de transporte, estações de captação, armazenamento e distribuição de água, serviços de emergência etc. Nesse sentido, o problema pode se agravar consideravelmente nos casos relacionados a esse tipo de infraestrutura e, dependendo das vulnerabilidades nelas existentes e da criticidade do ataque, tal prejuízo pode ser catastrófico. Um exemplo disso ocorreu recentemente na Flórida, Estados Unidos,

onde um *hacker* obteve acesso ao sistema de tratamento de água, aumentando a proporção de soda cáustica e expondo a população local ao risco de contaminação química [3]. Felizmente, um funcionário local percebeu o ocorrido e reverteu a ação em tempo oportuno.

Dessa maneira, a detecção de ataque cibernético tem assumido um papel protagonista em assuntos relacionados à prevenção e mitigação de ameaças no campo da tecnologia da informação.

2. Detecção de ataque

Ataques cibernéticos são manobras ofensivas e maliciosas direcionadas a sistemas de informação, infraestruturas computacionais, redes de dados e equipamentos computacionais pessoais. Possuem o objetivo de expor, alterar, desabilitar, destruir, roubar ou obter acesso não autorizado a dados ou recursos computacionais [4]. Dessa forma, um ataque cibernético compromete pelo menos um dos três aspectos da segurança da informação: confidencialidade, integridade ou disponibilidade [5].

A detecção do ataque é uma tarefa desempenhada por um Sistema de Detecção de Intrusão (IDS, do inglês *Intrusion Detection System*) – neste contexto, os termos ataque e intrusão serão utilizados indistintamente. Essa detecção ocorre por meio da monitoração de eventos que ocorrem em um sistema de computador ou rede de dados. Esses eventos são então analisados no intuito de encontrar sinais de possíveis incidentes, representando violações ou iminente ameaça de violação de políticas de segurança [6]. Conforme ilustrado na **Figura 1**, existem dois métodos de detecção básicos:

- Detecção baseada em assinatura: detecção de traços, denominados de assinatura, que identificam de forma única um determinado ataque. Esses traços são armazenados em um banco de dados que precisa ser constantemente atualizado, conforme novos tipos de ataques surgem.
- Detecção baseada em anomalia: detecção de padrões que estão fora daqueles considerados normais ou aceitáveis. Esse comportamento normal pode ser definido por um especialista de segurança em redes de dados ou pode ser aprendido por alguma técnica de Aprendizado de Máquina (AM).

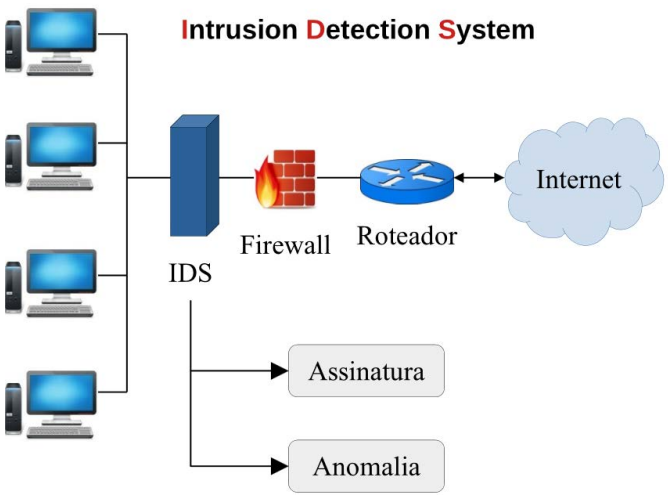

Fig. 1 – Exemplo de utilização de IDS. **Fonte: Adaptado de [7].**

O esquema na **Figura 1** mostra um IDS composto por apenas um equipamento em um local específico da rede, normalmente entre o *firewall* e a rede local a ser protegida. A vantagem desse esquema é que ele diminui consideravelmente a quantidade de dados maliciosos a ser analisada pelo IDS, uma vez que grande parte do tráfego considerado inapropriado é barrado no *firewall*. Além disso, instituições de maior porte costumam possuir uma rede local bastante extensa, muitas vezes composta por inúmeras sub-redes. Nesse caso, a implantação do IDS é mais complexa, exigindo a instalação de outros componentes, tais como:

- Sensores: equipamentos conectados em diferentes pontos da rede, para coletar dados, por exemplo, pacotes IP;
- Agentes: software com funções similares aos sensores, porém instalados em *hosts*. Dessa maneira, agentes monitoram computadores, podendo coletar outros dados além dos específicos de rede, por exemplo, acessos ao sistema de arquivo;
- Servidor de gerenciamento: responsável por receber, processar e correlacionar os dados enviados pelos sensores e agentes. Normalmente é neste equipamento onde é gerado o alerta de intrusão;
- Servidor de banco de dados: responsável por armazenar os dados coletados pelos agentes e sensores;
- Console: responsável por fornecer uma interface para os usuários e administradores do IDS.

Normalmente, o gerenciamento e os dados trocados entre esses equipamentos ocorrem em uma rede independente, o que proporciona maior proteção contra ataques dirigidos a um IDS. Essa rede independente pode ser, idealmente, uma segunda rede física, proporcionando mais segurança, porém, possui um custo de instalação maior. Uma alternativa é usar uma VLAN (do inglês *Virtual Local Area Network*) que compartilha a mesma rede física, ou ainda trafegar dados de gerenciamento sem o uso da VLAN, porém, com criptografia. Ambas as alternativas possuem um custo de instalação menor, embora aumentem o consumo de banda da rede de dados principal, além de reduzirem também a segurança dos equipamentos que compõem o IDS.

Independentemente do tamanho da rede e da quantidade de sensores e agentes, na prática, utilizam-se IDSs baseados em assinatura e anomalia em conjunto, visto que um tipo complementa o outro. Isso porque um IDS baseado em assinatura possui a vantagem de ter baixa Taxa de Falso Positivo (TFP), embora seja ineficiente na detecção de novos tipos de ataques. Essa ineficiência mantém-se mesmo quando esses novos ataques são apenas pequenas variações daqueles já catalogados. No caso de um IDS baseado em anomalias, ocorre o contrário: ele possui a vantagem de detectar novos ataques, porém, com uma tendência de maior TFP. Isso ocorre pois é difícil modelar precisamente o comportamento normal da rede, que pode variar consideravelmente dependendo do seu tamanho e da sua complexidade. Além disso, esse comportamento normal pode evoluir com o tempo, necessitando de revisões e atualizações do modelo que o representa.

Em resumo, o ideal é que o IDS forneça uma alta abrangência (também conhecida por sensibilidade, *recall*, Taxa de Verdadeiro Positivo (TVP) ou taxa de detecção), com uma baixa TFP. Essa característica é altamente desejável, sendo representada pela maior área possível abaixo da curva ROC (do inglês *Receiver Operating Characteristic*). Essa curva foi desenvolvida por operadores militares de radar, mostrando a relação de compromisso das características de TFP e abrangência do receptor do radar, o que explica sua nomenclatura. A **Figura 2** caracteriza as curvas ROC de três classificadores diferentes. Pontos distintos pertencentes a uma mesma curva significam limiares de decisão distintos para um mesmo classificador. Note que os melhores classificadores conseguem obter uma maior abrangência com baixa TFP e que isso resulta numa maior área sob a curva ROC. Além disso, a menor área ocorre quando o classificador é puramente aleatório. Nesse caso, onde a área é igual a 0,5, o classificador possui sempre a mesma probabilidade de detecção, independentemente da amostra a ser classificada.

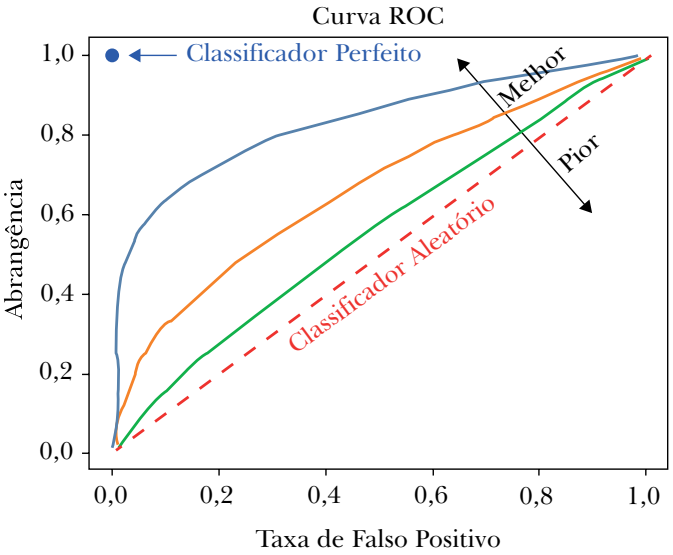

Fig. 2 – Exemplos de curva ROC**. Fonte: Adaptado de [8].**

3. Aprendizado de Máquina

O AM é um campo da Inteligência Artificial que está fortemente relacionado à estatística computacional, focando na atividade de predição por meio de métodos computacionais de otimização [9], o que possibilita a execução de tarefas de classificação ou regressão pelo computador. Classificar significa determinar a qual classe uma determinada amostra pertence. Como existe uma quantidade finita de classes, estas podem ser representadas por valores discretos. Um exemplo seria classificar e-mails em *spam* ou não-*spam*. Já na tarefa de regressão, a amostra é relacionada a um valor contínuo, como, por exemplo, predição de valores de imóveis.

Tais amostras são representadas por suas características importantes, também chamadas de atributos. No caso de uma classificação em *spam*, por exemplo, cada e-mail é considerado uma amostra. Um possível atributo pode ser o número de vezes que determinada palavra, ou composição de palavras, aparece no e-mail. Exemplos de tais palavras são: incrível, satisfação, agora, bônus, ganhar, oferta, desconto etc. Já no caso dos imóveis, alguns exemplos de atributos são a área, o número de quartos, a localização e a idade.

Dessa maneira, algoritmos computacionais "aprendem", de forma indutiva, relações entre atributos existentes em uma base de dados [10]. As duas principais formas de aprendizado são: supervisionada e não supervisionada. No primeiro caso, são fornecidos exemplos, compostos por atributos e respostas, para que o modelo aprenda, de forma indutiva e aproximada, a função alvo. Essa função representa o mapeamento entre os atributos e as respostas (também conhecidas por atributo alvo ou rótulos), sendo normalmente complexa e desconhecida [11]. Classificação de e-mails em *spam* e predição de valores de imóveis são tarefas supervisionadas, uma vez que os valores-alvo (*spam*, não-*spam* e preço do imóvel) são fornecidos. Já no aprendizado não supervisionado, não há o fornecimento de rótulos ao algoritmo. Nesse caso, o objetivo do modelo é aprender, por si só, alguma estrutura inerente à base de dados [11]. Por exemplo, existem técnicas de agrupamento de amostras baseadas em métricas de similaridade, conhecidas por clusterização. Técnicas de projeção, como Análise de Componentes Principais (PCA, do inglês *Principal Component Analysis*), também pertencem a essa categoria. Tais técnicas objetivam o agrupamento e a redução da dimensionalidade dos dados, dois pilares do aprendizado não supervisionado [12].

Para uma efetiva aplicação de AM, a base de dados é dividida em três conjuntos: treinamento, validação e teste. No conjunto de treinamento, algoritmos computacionais são aplicados de forma iterativa, com o objetivo de minimizar uma função conhecida por custo. Essa função indica se os valores preditos a partir dos atributos estão, em média, próximos daqueles contidos nos rótulos [11]. Quanto menor o valor da função custo, maior será essa proximidade. Uma técnica bastante utilizada nessa minimização é o gradiente descendente, que ajusta os parâmetros do modelo de forma automática, com base no gradiente da função custo em relação a esses parâmetros.

O conjunto de teste é utilizado para verificar o desempenho do modelo em exemplos diferentes do conjunto de treino, indicando se a relação entre atributos e rótulos foi aprendida de forma adequada. Nesse caso ideal, o modelo consegue generalizar bem essa relação, sendo capaz de desconsiderar possíveis ruídos inerentes ao conjunto de treinamento.

Quando não ocorre essa situação ideal, é provável que o modelo esteja com um alto viés ou superajuste. Embora ambos degradem o desempenho no conjunto de teste, suas causas são bastantes distintas. Alto viés ocorre quando o modelo é muito simples, incapaz de aproximar a função alvo, independentemente de qualquer ajuste em seus parâmetros. Tal problema é verificado quando o desempenho é baixo tanto no conjunto de treino quanto no de teste. Para resolvê-lo, é necessário substituir o modelo por outro mais complexo, podendo ser também necessário obter uma quantidade maior de atributos. Por outro lado, o superajuste ocorre quando o modelo é capaz de aproximar funções mais complexas que a própria função alvo [13]. Consequentemente, o modelo acaba gerando uma função que se ajusta demasiadamente ao conjunto treino, influenciada por ruídos e imperfeições, que não generalizam a função alvo adequadamente [10, 14]. Esse problema é verificado quando há um desempenho consideravelmente superior no conjunto de treino em comparação ao conjunto de teste. O superajuste pode ser reduzido com a obtenção de uma quantidade maior de exemplos de treinamento, ou, ainda, por técnicas de regularização.

Além dos parâmetros do modelo, que são ajustados de forma automática por algoritmos de otimização, existem outros que precisam ser ajustados de forma manual e empírica, denominados de hiperparâmetros. Alguns exemplos de hiperparâmetros são: profundidade da árvore de decisão, número de neurônios e camadas da rede neural, taxa de

aprendizado, taxa de regularização etc. A principal função do conjunto de validação é auxiliar no ajuste dos hiperparâmetros [15]. Estes recebem diferentes valores, que são utilizados para treinar o modelo no conjunto de treino, e, então, validados no conjunto de validação, onde são selecionados os hiperparâmetros que obtiveram o melhor desempenho. Tal procedimento pode ser realizado por meio de uma Busca em Grade (*grid search*).

Uma aplicação de AM que vem ganhando espaço no meio acadêmico é a detecção e classificação de ataques cibernéticos. Isso se deve principalmente à disponibilização de bases de dados públicas, conforme apresentado na seção 4.

4. Base de dados com tráfego anômalo

Bases de dados são essenciais para o AM, pois contêm a informação a ser aprendida de forma indutiva pelo modelo. Entretanto, existe uma grande ausência de bases de dados com qualidade, obtidas por coleta de tráfego real. Uma razão para isso é que empresas podem acabar expondo alguma vulnerabilidade ou informação sensível ao disponibilizar dados para pesquisa sobre uma eventual invasão sofrida. Outra razão se dá por ser complicado obter os rótulos de tráfegos reais. Um *hacker* provavelmente não teria a boa vontade de informar à vítima qual foi a classe do ataque empregado. Na verdade, ele sequer vai informar que fez um ataque. Uma alternativa é obter os rótulos por meio de um IDS de assinatura; entretanto, ataques desconhecidos (ainda sem assinatura) seriam rotulados como tráfego normal, influenciando negativamente o aprendizado do algoritmo.

Dessa forma, não é por acaso que praticamente todas as bases de dados públicas de ciberataque foram criadas por meio de simulação. Um fator importante a ser considerado nessas bases é o nível de semelhança em relação a uma situação real. Provavelmente, um IDS com alto desempenho em uma base não realista poderá não funcionar adequadamente em um emprego real.

Outro aspecto relevante se refere aos exemplos constantes nessas bases. Eles podem ser fornecidos em dados brutos (na forma de pacotes IP), ou em dados processados (na forma de tráfego ou fluxo de rede). No primeiro caso, cada pacote IP representa um exemplo da base de dados. Alguns atributos são comuns a todos os pacotes, como, por exemplo, tamanho do cabeçalho, tamanho total do pacote, protocolo, endereço IP de origem e destino. Outros atributos são específicos do protocolo (TCP, UDP etc.) como, por exemplo, portas de origem e destino, que não existem em todos os protocolos. Esses pacotes podem ser capturados e armazenados no formato pcap, por meio de aplicações específicas. Algumas aplicações populares são o TCPDump, o Wireshark, o Snort e o Nmap [16].

Já na forma de fluxo de rede, esses pacotes IPs são processados e formatados de maneira que cada exemplo da base de dados consista em informações que definem uma sequência uni ou bidirecional de pacotes que compartilham os seguintes atributos: endereço IP de origem, endereço IP de destino, protocolo, porta de origem e porta de destino [16]. Conexões TCP são um exemplo de fluxo de dados bidirecional. Além disso, outros dados estatísticos podem se somar a esses atributos, como, por exemplo, quantidade de bytes emitidos pela origem, quantidade de bytes emitidos pelo destino, tempo médio entre pacotes, taxa de pacotes etc. Em comparação com bases de dados na forma de pacotes IP, bases de dados de fluxo de rede possuem um tamanho menor, pois desconsideram informações no *payload* dos pacotes.

A partir do ano de 1998, começaram a ser disponibilizadas bases de dados de intrusão, o que possibilitou a introdução de técnicas de AM no campo da segurança cibernética. Essas bases foram geradas por meio de simulação de tráfegos maliciosos e benignos. Algumas delas também incluíram tráfego de fundo, ou seja, tráfegos reais e anonimizados, os quais não se sabe precisamente se são benignos ou malignos. Em alguns casos, também são fornecidas as classes do tráfego maligno. As principais bases públicas disponíveis são:

- DARPA 1998/1999 [17]: *Defense Advanced Research Projects Agency* DARPA 1998 e DARPA 1999, criadas pelo Instituto de Tecnologia de Massachusetts (MIT), sendo amplamente utilizadas

e discutidas em diversos artigos. A primeira foi obtida em uma simulação que durou nove semanas. As primeiras sete semanas deram origem ao conjunto de treinamento, e as duas últimas, ao conjunto de teste. Os atributos dessa base são fornecidos na forma de dados brutos, ou seja, pacotes IP. Um ano após, houve a disponibilidade de uma segunda base, intitulada de DARPA 1999. Ela foi gerada por uma simulação de cinco semanas, sendo as três primeiras de treinamento e as duas últimas para teste. Essa base possui uma quantidade significativamente maior de tipos de ataques em relação à primeira. Os atributos dessa base também são fornecidos por meio de pacotes IP.

- KDD Cup 1999 [18]: utilizada em uma competição de classificação de ataques cibernéticos, essa base possui 41 atributos orientados a fluxo de rede (NetFlow), sendo derivada da DARPA 1998. A base KDD Cup 1999, entretanto, possui sérias limitações [16] como, por exemplo, alta redundância de amostras e imprecisões derivadas de perda de pacotes durante sua criação, causada por excesso de tráfego de dados. Além disso, a grande maioria dos exemplos, tanto no conjunto de treino, como no conjunto de teste, era de fácil classificação, não exigindo modelos muito complexos [19].
- NSL-KDD [20]: foi construída em 2009 por uma amostragem cuidadosa da base KDD 1999, onde foram eliminadas as redundâncias e reduzido o número de exemplos fáceis. Dessa forma, altas acurácias não eram mais possíveis de serem obtidas com modelos demasiadamente simples. Os principais ataques simulados nessa base são DoS (Negação de Serviço), acesso não autorizado de administrador (U2R – *User to Root*), acesso a um *host* da rede local por uma máquina remota não autorizada (R2L – *Remote to Local*) e varredura de informações de recursos da rede (*Probe* ou *Scan*).
- CTU-13 [21]: foi criada em 2011 pela Universidade de CTU, República Tcheca. O termo 13 advém dessa base possuir treze diferentes cenários onde ocorrem ataques de *Botnet*. Tais cenários variam bastante em nível de dificuldade de detecção de ataque. Os dados dessa base são fornecidos em dados brutos (pacotes IP) e dados processados (padrão NetFlow, WebLogs e outros). Delplace, Hermoso e Anandita [22] utilizaram essa base para verificar o desempenho de vários modelos de AM, além de analisar a importância relativa de cada atributo.
- CIC-IDS [23]: são bases públicas disponibilizadas pelo Instituto Canadense de Cibersegurança, localizado na Universidade de New Brunswick. Existem três: uma lançada em 2012, uma, em 2017, e outra, mais recente, em 2018. As duas últimas contêm tráfego benigno e ataques mais atuais. Os dados estão disponíveis na forma bruta (pcap) e na forma de fluxo de rede, através do processamento dos dados brutos por uma ferramenta própria desse Instituto, chamada de CICFlowMeter. Essa ferramenta caracteriza o fluxo de rede através de seis atributos: *time stamp*, endereço IP de origem e destino, porta de origem e destino e protocolo. Uma característica interessante nessa base foi o método para gerar o tráfego benigno, que imita o comportamento de interações humanas, portanto sendo mais parecido com o tráfego real.
- AWID [24]: é um projeto da Universidade de Aegean, localizada na Grécia, onde são disponibilizadas duas bases de dados públicas rotuladas (AWID2 e AWID3), extraídas de tráfego Wi-Fi. Elas possibilitam desenvolver e analisar IDSs específicos para o ambiente sem fio, que apresenta vulnerabilidades diferentes do cabeado. A AWID2, disponibilizada em 2015, é considerada a primeira base pública de redes sem fio [25], e possui três categorias de ataque: injeção, negação de serviço e personificação, além de tráfego normal. Esses ataques foram projetados para explorar vulnerabilidades existentes no ambiente de autenticação, que utiliza WEP (*Wired Equivalent Protection*), WPA e WPA2 (*Wi-Fi Protected Access*). Já a AWID3, lançada em 2021, utilizou um ambiente de autenticação mais moderno, denominado EAP (do inglês *Extensible Authentication Protocol*), possibilitando o estudo de ataques projetados para explorar vulnerabilidades nesse novo ambiente. Também foram incluídos ataques multicamadas, que exploram vulnerabilidades da camada de *link* (802.11) e das camadas superiores.

- UNSW-NB15 [26]: é uma base pública de ataques cibernéticos lançada em 2015, disponível nos formatos pcap, BRO, Argus e csv. Os exemplos dessa base foram criados por meio de uma ferramenta intitulada IXIA *Perfect Storm*, pertencente ao Centro Australiano de Cibersegurança (ACCS, do inglês *Australian Center for Cyber Security*). Essa base possui nove tipos de ataque, disponibilizados na forma de 49 atributos, além do rótulo da classe. Está disponível na versão completa com um total de 2.540.044 exemplos, e, na versão reduzida, com os conjuntos de treino e de teste possuindo 175.341 e 82.332 exemplos, respectivamente.
- IoT-23 [27]: é uma base que contém tráfego capturado de 2018 a 2019, e foi publicada em janeiro de 2020, na Universidade de CTU, República Tcheca, sendo financiada pela empresa de antivírus Avast. O termo 23 advém dessa base possuir 23 diferentes cenários onde ocorrem ataques relacionados a equipamentos do tipo IoT (do inglês *Internet of Things*). É uma base rotulada, disponibilizada na forma de arquivos pcap e na forma de tráfego de dados, obtidos por meio do *software* Zeek. Possui oito tipos de ataques, além do tráfego benigno, que foi gerado por três equipamentos IoT sabidamente não infectados.
- USB-IDS-1 [28]: é uma base disponibilizada em 2021, pela Universidade de Sannio, Benevento (USB), Itália, constituída por uma variedade de ataques DoS (Hulk, TCPFlood, Slowloris e Slowhttptest) executados contra um servidor de páginas. Ela pode ser obtida por meio de dados brutos (formato pcap) ou de tráfego de dados (formato csv), este último gerado pela mesma ferramenta utilizada nas bases CIC-IDS: CICFlowMeter. Por esse motivo, os exemplos existentes na USB-IDS-1 possuem os mesmos atributos daquelas bases, permitindo testar a capacidade de generalização de modelos de AM em bases distintas. Um fato inovador na base USB-IDS-1 foi a inclusão de ferramentas e configurações defensivas no servidor de páginas, o que é muito comum em aplicações reais. Tais ferramentas alteram o perfil do tráfego, podendo diminuir a eficácia de algoritmos treinados em outros ambientes. Além disso, os autores mostraram que essas ferramentas não garantem uma perfeita mitigação dos ataques, o que reforça a necessidade da utilização de IDS.

5. Explicabilidade

Recentemente, métodos que utilizam aprendizado profundo, como, por exemplo, redes convolucionais e recorrentes, vêm obtendo expressiva acurácia em tarefas de classificação de imagem e processamento de linguagem natural. Essa evolução foi aproveitada na área de detecção de intrusão, possibilitando uma elevada acurácia na detecção de ciberataques, com reduzida TFP [29]. Por outro lado, tais modelos são pouco transparentes, o que tem aumentado a importância no desenvolvimento de métodos que explicam de forma razoável as razões por trás das decisões tomadas por esses modelos. Entretanto, a grande maioria dos artigos nessa área ainda se preocupa exclusivamente com a obtenção de elevada acurácia, altas taxas de detecção e baixas TFPs, sem levar em consideração o entendimento dos mecanismos que levaram o modelo a realizar determinada classificação. Com o surgimento de bases de dados públicas, houve uma tendência a uma certa competição entre os autores de artigos dessa área. É muito comum encontrar frases do tipo: "O nosso método obteve uma acurácia superior, com uma menor taxa de falso positivo". Também é muito comum, nesses artigos, a exibição de uma tabela listando o desempenho de vários outros métodos, extraídos de outros trabalhos, normalmente com desempenho inferior ao proposto.

Essa busca por um alto desempenho resultou em um aumento considerável na complexidade dos modelos, o que tornou praticamente impossível entender o mecanismo por trás da decisão desses algoritmos nas tarefas de detecção de intrusão. Isso não significa a inexistência de conhecimento a respeito de como o algoritmo realiza a classificação. Por exemplo, em redes MLP (do inglês *Multi Layer Perceptron*), também conhecidas por redes neurais, por mais profundas e complexas que sejam, as decisões do algoritmo continuam a ser geradas por uma intrincada combinação ponderada de funções camada a camada. O termo

"ponderada" se refere aos pesos que multiplicam os atributos e os valores de cada neurônio. Esses pesos são ajustados de forma automática, normalmente pela técnica do gradiente descendente ou alguma variação dela, de maneira a minimizar a diferença entre a classe predita e o rótulo a que de fato pertence cada exemplo da base de dados. O problema é que isso, por si só, não esclarece quais atributos e quais relações entre esses atributos levaram a uma determinada classificação. Para o algoritmo, o que importa é minimizar o erro, independentemente de qualquer racionalidade em relação às características dos exemplos.

Essa falta de transparência nos modelos complexos causa desconfiança especialmente em classificações não óbvias, que possam contrariar a opinião de um humano. Nesse caso, se a decisão do modelo não informa suas razões, fica difícil saber se houve erro ou se o modelo foi capaz de perceber algum detalhe importante que escapou à percepção humana. Esse problema é mais relevante em aplicações com baixa tolerância a falhas, como ocorre na detecção de uma intrusão. Nesse caso, há uma baixa tolerância a falsos negativos, pois nenhuma ação oportuna será tomada para interromper um ataque não detectado, cuja presença só será notada pelas suas consequências. Um modelo "opaco" não permite ao analista saber quando e por que determinados ataques não são detectados. A aplicação em intrusão de redes é bem distinta de, por exemplo, sistemas de recomendação. Nesse caso, não há tanta importância em se saber porque o modelo recomenda determinado produto a um usuário, desde que o algoritmo possua uma elevada acurácia. Esses casos em que o modelo erra, que são minoria, não trazem graves consequências, apenas fazendo com que o usuário desconsidere tais recomendações.

Além disso, o entendimento de decisões corretas tomadas por modelos complexos pode ajudar o especialista a descobrir relações não triviais entre os atributos e os ataques cibernéticos até então desconhecidas. Por outro lado, o entendimento de decisões errôneas possibilita o aperfeiçoamento do modelo. Finalmente, atacantes podem realizar mudanças específicas em seus ataques de maneira a se evadirem da detecção. Assim, o conhecimento do mecanismo de classificação permite ao especialista realizar adaptações que robusteçam o modelo contra essa vulnerabilidade.

Consequentemente, surgiram novos estudos e novas ferramentas com o intuito de abrir essa caixa-preta e prover um entendimento mais racional dos fatores mais relevantes na tomada de decisão desses modelos. Esses estudos e ferramentas provêm o que se convencionou chamar de explicabilidade do modelo. Dessa forma, os trabalhos mais recentes que tratam sobre detecção de intrusão usando AM começaram a abordar não somente o desempenho do modelo, mas também a sua explicabilidade, permitindo ao analista verificar a razoabilidade das relações entre os atributos abstraídas pelo modelo na atividade de classificação. A **Figura 3** ilustra esse procedimento.

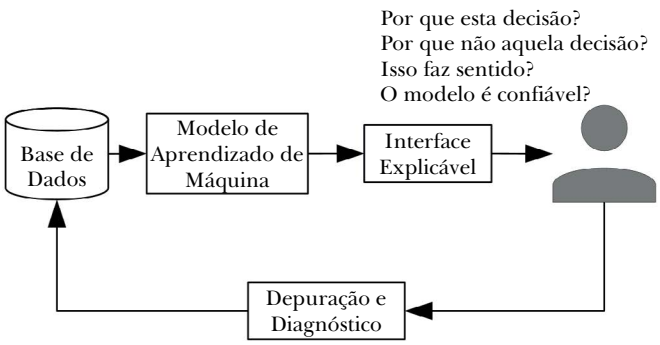

Fig. 3 – Interface explicável**. Fonte: Adaptada de [30].**

6. Trabalhos em IDS

Devido à importância desse assunto, foram publicados diversos artigos de revisão sobre detecção de intrusão utilizando AM. Alguns, com destacado número de citações [16, 31] sintetizam a maioria dos objetivos de revisão, além de oferecerem um relevante arcabouço teórico e de contribuírem com melhorias na divulgação e análise de resultados. Adicionalmente, Buczak e Guven [16] ressaltaram a variedade de técnicas de AM disponíveis para aplicação em IDS, mostrando como é complexo estabelecer qual delas seria a mais apropriada, levando-se em consideração o tipo de ataque que o sistema deverá detectar. Outra contribuição importante, apresentada por Khraisat *et al.* [31], é o estudo de técnicas de evasão

utilizadas por atacantes, um dos principais desafios à pesquisa de detecção de intrusão. Embora tais aspectos sejam essenciais, esses artigos de revisão não incluem trabalhos de análises por meio da explicabilidade, o que só pode ser encontrado em artigos de revisão de áreas distintas da cibernética. Uma segunda lacuna se refere à inclusão de trabalhos que abordam o potencial que modelos de IDS têm na geração de falsos positivos em aplicações reais. Assim, esta pesquisa procura complementar os artigos de revisão já existentes por meio da inclusão e análise desses dois tópicos. Dessa forma, a subseção 6.1 relaciona os trabalhos que utilizam AM, privilegiando o desempenho do algoritmo, porém, não apresentam uma análise crítica de outros fatores relevantes, como, por exemplo, os impactos relativos à TFP, ou, ainda, a explicabilidade do modelo. Tais fatores foram analisados pelos trabalhos constantes nas subseções 6.2 e 6.3, respectivamente. Finalmente, a subseção 6.4 apresenta um comparativo entre esses trabalhos.

6.1. Trabalhos com foco no desempenho

Buczak e Guven [16] reuniram trabalhos que procuram desenvolver IDS usando detecção por assinatura e por anomalia. Esta última tem despertado um maior interesse da comunidade científica por empregar técnicas clássicas de AM e mineração de dados. Também são de maior interesse os trabalhos que utilizaram bases públicas e que apresentaram alguma informação relativa ao desempenho. Deve ser levado em consideração que, nas bases DARPA e derivadas (à exceção da NSL-KDD), é comum serem obtidos resultados com desempenho muito elevado, fora do contexto de aplicações reais. Isso ocorre devido a uma grande quantidade de registros repetidos, além de uma predominância de ataques de fácil classificação [19]. Por outro lado, Khraisat *et al*. [31] realizaram uma pesquisa mais atual na área de detecção de intrusão, incluindo trabalhos que utilizaram bases mais recentes, em especial a CIC-IDS 2017. Os trabalhos mais relevantes dessas pesquisas constam na **Tabela 1**, sendo brevemente descritos no decorrer da subseção 6.1.

Tab. 1 – Trabalhos citados em artigos de pesquisa.

Artigos de Pesquisa	Trabalhos Relevantes
Buczak e Guven [16]	Fan *et al*. [32] Hu, Liao e Vemuri [33] Bivens *et al*. [34] Kruegel *et al*. [35] Shon e Moon [36] Tajbakhsh, Rahmati e Mirzaei [37] Amor, Benferhat e Elouedi [38]
Khraisat *et al*. [31]	Chen, Hsu e Shen [39] Adebowale, Idowu e Amarachi [40] Thaseen e Kumar [41] Ustebay, Turgut e Aydin [42]

Fonte: Elaborado pelos autores.

Fan *et al*. [32] usaram uma técnica de aprendizado indutivo de criação de regras intitulado RIPPER (do inglês *Repeated Incremental Pruning to Produce Error Reduction*). Os autores desenvolveram um gerador artificial de anomalias (ataques) para melhorar a habilidade de generalização do classificador. Foi utilizada a base DARPA 1998, obtendo-se uma abrangência de 94%, com uma TFP de 2%. Ainda nessa mesma base, Hu, Liao e Vemuri [33] utilizaram uma variação das máquinas de vetores de suporte, intitulada de RSVM (do inglês *Robust Support Vector Machine*). Essa técnica obtém uma média de hiperplanos discriminantes, suavizando a fronteira de classificação, além de obter o parâmetro de regularização e o classificador de forma automática. Foi obtido um bom desempenho mesmo na presença de ruído (apesar de alguns rótulos da base de treino estarem incorretos): 75% de abrangência sem alarmes falsos, e 100% de abrangência com uma TFP de 3%.

Bivens *et al*. [34] utilizaram SOM (do inglês *Self-Organizing Map*), para aprender o comportamento normal do tráfego, em conjunto com MLP, para classificar intrusões na base DARPA 1999. Os autores procuraram classificar ataques em diferentes categorias e relatam ter obtido 100% de TVN (Taxa de Verdadeiro Negativo) e uma TFP de 76%. Essa informação é um tanto inconsistente, uma vez que uma TVN de 100% implica em uma TFP igual a zero. Na verdade, o que os autores chamam de TFP é a proporção de ataques classificada de forma errônea em um outro tipo de ataque. Nessa mesma base, Kruegel *et al*. [35] utilizaram redes bayesianas, obtendo uma TFP ligeiramente maior que 0,2% e

uma abrangência de 100%. Por outro lado, Shon e Moon [36] utilizaram uma técnica de detecção de anomalias denominada SVM aprimorada (do inglês, *Enhanced* SVM), para detecção de ataques na base DARPA 1999. Essa técnica foi derivada da *One-Class* SVM e da *Soft-Margin* SVM. Para uma aplicação mais realista, o desbalanceamento da base foi aumentado, consistindo-se em 1% a 1,5% de ataques e 98,5% a 99% de tráfego normal. Obteve-se uma taxa de falso negativo (TFN) de 27,27%, com uma TFP de 10,2%. Embora a abrangência não tenha sido diretamente fornecida, é possível derivá-la da TFN, sendo, portanto, 72,73%.

Tajbakhsh, Rahmati e Mirzaei [37] utilizaram Mineração de Regras de Associação Fuzzy para encontrar padrões nos relacionamentos dos atributos da base KDD 1999. O artigo destaca alguns benefícios da técnica, como a criação de regras humanamente interpretáveis; entretanto, obteve uma TFP de 13% a uma abrangência de 100%. Ainda nessa mesma base, Amor, Benferhat e Elouedi [38] utilizaram o algoritmo Naive Bayes (NB), reportando uma abrangência de 89% e uma TVN de 98%. Embora não tenha sido reportada a TFP, pode-se inferir, baseado na TVN, que ela é de 2%.

Chen, Hsu e Shen [39] extraíram dados de chamadas de sistema da base DARPA 1998, codificando-os com o método tf-idf (do inglês *term frequency – inverse document frequency*), para, então, treinar um classificador SVM. Foi obtida uma abrangência de 100% com uma TFP de 8,53%.

Adebowale, Idowu e Amarachi [40] utilizaram SVM, MLP, NB e Árvores de Decisão (DT) para a detecção de intrusão na base NSL-KDD, obtendo as acurácias de 97,3%, 95,8%, 89,6% e 99,6%, respectivamente. O autor utilizou o Weka 3.6.7 (Waikato *Environment for Knowledge Analysis*), um software desenvolvido em Java pela Universidade de Waikato, Nova Zelândia, que incorpora várias técnicas de AM. Não foi informado se os hiperparâmetros dos modelos foram os valores padrão, ou se houve algum ajuste para obter tais acurácias. Estas foram levantadas por meio de uma validação cruzada tipo 10-*fold* na base "NSL-KDD completa". Embora a base NSL-KDD seja fornecida em três conjuntos, 20% treino (apenas vinte por cento das amostras do conjunto treino), treino e teste, o autor não esclarece o que vem a ser a "NSL-KDD completa". Uma possibilidade, portanto, é a de que ele juntou as partes de treino e teste numa única base e aplicou validação cruzada, somente com o objetivo de treinar e avaliar o desempenho dos modelos (e não de ajustar hiperparâmetros). As acurácias obtidas ficaram bem acima daquelas levantadas por Tavallaee *et al*. [19], que foram os criadores da base NSL-KDD. Provavelmente, a razão para isso foi que, neste último artigo, a base de treino usada foi de 20% do treino, e as acurácias foram levantadas no conjunto teste. Adebowale, Idowu e Amarachi [40] também forneceram os índices de detecção e TFP: o SVM obteve uma abrangência de 95,9% a uma TFP de 1,4%; já o MLP, o NB e o DT obtiveram, respectivamente, uma abrangência de 95,9% a uma TFP de 4,4%, uma abrangência de 87,7% a uma TFP de 8,8%, e uma abrangência de 99,6% a uma TFP de 0,4%.

Thaseen e Kumar [41] utilizaram a base NSL-KDD para testar todos os classificadores baseados em árvore, constantes no software Weka, sendo eles: ADTree, C4.5, J48graft, LADTree, NBTree, RandomTree, RandomForest e REPTree. Um pré-processamento foi realizado na base de dados, onde os atributos contínuos foram discretizados por uma técnica de filtragem supervisionada de atributos chamada *Discretize*. Além disso, houve uma redução de 41 para apenas 8 atributos, obtidos via análise CFS (do inglês *Correlation Feature Selection*). Trata-se de um método de seleção de atributos que privilegia aqueles com maior correlação com a variável de saída e, ao mesmo tempo, menor correlação com os demais atributos. Embora não tenha havido um ajuste de hiperparâmetros, em geral, os classificadores obtiveram elevada acurácia no conjunto de teste. NBTree foi o que obteve o melhor desempenho com uma abrangência de 97,8% a uma TFP de 4,8%.

Ustebay, Turgut e Aydin [42] obtiveram uma considerável redução nos atributos dos exemplos da base de dados CIC-IDS 2017, utilizando a eliminação recursiva de atributos com o algoritmo

Floresta Aleatória (RF, do inglês *Random Forest)*. Assim, a quantidade total de atributos foi reduzida de 81 para apenas quatro, selecionados de acordo com a sua importância. Os autores levantaram um ponto relevante ao afirmar que os atributos *Flow_ID*, *Source_IP*, *Destination_IP*, *Timestamp* e *External_IP* não foram utilizados no treinamento, ou seja, nenhum deles estava presente nos quatro selecionados. Isso é um indício de que o classificador poderá ter um desempenho razoavelmente próximo ao ser aplicado em uma rede real, pois tais atributos são específicos da base CIC-IDS 2017 e não deveriam ser considerados na tarefa de classificação. Entretanto, não houve uma análise crítica do quão geral é o significado de cada um dos quatro atributos selecionados, ou seja, se eles se mantêm da mesma forma em redes reais. Após a seleção de atributos, a base foi particionada em 80% para treino e 20% para teste. Em seguida, utilizou-se uma MLP de três camadas com funções de ativação do tipo ReLu, com exceção da camada de saída, cuja função de ativação foi a Estimação de Momento Adaptativo. Ao analisar a curva ROC, é possível inferir uma TFP de 18% a uma abrangência de 100%.

Há também trabalhos que utilizaram redes neurais recorrentes na tarefa de detecção de intrusão. Le, Kim e Kim [43] utilizaram o algoritmo de predição de sequência LSTM (do inglês *Long Short Term Memory*) para realizar a classificação de ataques em DoS, Varredura, U2R e R2L. A base de dados empregada foi a KDD 1999, sendo que o resultado obtido foi uma abrangência de 98,95%, com uma TFP de 9,98%. Entretanto, os autores não detalham o método de aplicação do algoritmo.

Xu *et al*. [44] realizaram um trabalho similar ao de Le, Kim e Kim [43], porém, substituindo o algoritmo LSTM por uma Rede Neural Recorrente de predição de sequência denominada GRU (do inglês *Gated Recurrent Units*). Os autores sugeriram que GRU é superior ao LSTM para classificar ataques, e exemplificaram tal comparação através de uma aplicação isolada de ambos os algoritmos nas bases de dados KDD 1999 e NSL-KDD. Nas duas bases, o GRU realmente foi superior ao LSTM, embora não haja suporte teórico para generalizar tal superioridade em outras aplicações. Além disso, tais algoritmos foram utilizados em conjunto com uma Rede Neural MLP. Os melhores resultados, obtidos com o GRU, foram uma abrangência de 99,42%, com uma TFP de 0,05%, para a base KDD 1999. Já no caso da base NSL-KDD, a abrangência foi de 99,31%, com uma TFP de 0,84%. Embora Le, Kim e Kim [43] tenham sido citados nas referências bibliográficas de Xu *et al*. [44], em nenhum momento os autores ressaltam o fato de terem obtido uma TFP extremamente reduzida, se comparada à do primeiro trabalho. Além disso, novamente, não há um detalhamento da metodologia de aplicação do algoritmo, onde poderiam ser apresentadas as sequências de atributos utilizadas, já que se trata de um algoritmo de predição sequencial.

Papamartzivanos [45] desenvolveu uma metodologia de indução de regras baseada numa combinação de algoritmos genéticos (GA, do inglês *Genetic Algorithm*) com árvores de decisão. Esse método, denominado *Dendron*, objetiva evoluir uma população de árvores de decisão, resultando num conjunto de regras para detecção e classificação de ataques. Segundo o autor, essa metodologia permite desenvolver um IDS baseado em assinatura com um desempenho balanceado, ou seja, com acurácia similar em todas as classes (inclusive as mais raras) nas quais os tráfegos de rede são classificados. Outra vantagem desse método é que, como ele utilizou árvores de decisão, as regras geradas para a classificação são humanamente inteligíveis, facilitando a tomada de decisão por especialistas em relação às contramedidas. O modelo foi testado nas bases KDD 1999, NSL-KDD e UNSW-NB15, obtendo, respectivamente, as seguintes métricas de desempenho: abrangência de 98,24%, com uma TFP de 0,75%; abrangência de 95,97%, com uma TFP de 1,08%; e abrangência de 63,76%, com uma TFP de 2,61%.

Ainda nesse mesmo trabalho, Papamartzivanos [45] baseou-se numa metodologia desenvolvida por Raina *et al*. [46], denominada STL (do inglês *Self-Taught Learning*), com o objetivo de tornar o IDS autoadaptável, permitindo que ele mantenha o desempenho mesmo na ocorrência de mudanças significativas no ambiente onde ele opera. Essas mudanças podem ter diferentes

causas, como a inserção e retirada de ativos da rede, inclusive equipamentos IoT, atualizações de software e surgimento de novos ataques. Portanto, diferente do que ocorre em IDSs de assinatura comuns (estáticos), não há a necessidade de atualizações manuais para adaptação a essas mudanças. Para demonstrar essa capacidade, o autor treinou um IDS numa base inicial contendo 10% de tráfego normal da base KDD 1999 e uma pequena parcela de ataques de cada classe dessa mesma base. O algoritmo utilizado foi uma rede DSAN (do inglês *Deep Sparse Autoencoder Network*) em conjunto com a regressão Softmax. O restante da base KDD 1999 foi utilizado para gerar 100 ambientes diferentes, por meio de amostragem aleatória. Cada ambiente poderia conter, ou não, classes ou instâncias pertencentes à base inicial; portanto, dependendo dessas diferenças, um determinado ambiente poderia ser pouco ou muito distinto da base inicial. Assim, partindo do IDS treinado, foram obtidos dados de desempenho nesses ambientes variados de duas formas: a primeira, mantendo o IDS estático, ou seja, sem alterá-lo após o treinamento inicial; e a segunda, utilizando o STL, para testar a capacidade do classificador em adaptar-se a mudanças. O IDS estático obteve, na média (considerando 100 ambientes diferentes), uma abrangência de 38,54%, bem inferior a 60,34%, que foi obtida pelo IDS autoadaptável. Um aspecto discordante nesse trabalho é que o experimento foi realizado em um IDS baseado em AM, utilizando DSAN e Softmax, sendo que o autor argumenta que essa técnica permite tornar um IDS baseado em assinatura autoadaptável.

Stefanova [47] desenvolveu um IDPS (do inglês *Intrusion Detection and Prevention Systems*), o qual possui duas camadas: a primeira, responsável por realizar a detecção por meio de AM, e a segunda, responsável por impedir ativamente o ataque detectado, por meio de Aprendizado por Reforço. Neste caso, o autor aplica uma aproximação do *Q-learning* no contexto de Teoria dos Jogos, onde o atacante e o defensor (da rede de dados) participam de um mesmo jogo cuja solução é um ponto de equilíbrio ótimo, do ponto de vista da defesa. A base utilizada foi a NSL-KDD. Com relação à primeira camada, houve uma seleção de atributos utilizando ganho de informação, reduzindo-os de 41 para 30. A classificação foi realizada por uma RF, obtendo aproximadamente uma TFP de 0,16% e uma abrangência de 99,9%. Tais valores foram obtidos por inspeção da curva ROC presente no trabalho.

6.2. Trabalhos com foco na redução da TFP

Os trabalhos a seguir abordaram o problema da TFP, que é um dos principais obstáculos na implantação de um IDS em redes reais, principalmente naquelas com alta densidade de tráfego.

Subba, Biswas e Karmakar [48] propuseram um método de redução da TFP em IDS de assinatura, cuja ideia principal consiste em escanear todos os ativos pertencentes à rede local a ser protegida e levantar todas as vulnerabilidades existentes, criando um "perfil de vulnerabilidades". Dessa forma, todos os alarmes referentes a ataques que exploram vulnerabilidades inexistentes nesse perfil são descartados, reduzindo consideravelmente a TFP, sem impactar na abrangência. O experimento utilizou o IDS Snort, cujo banco de dados de assinatura estava na versão Snort V2.8 com certificação VRT (do inglês *Vulnerability Research Team*), na configuração padrão com todas as assinaturas habilitadas. Levantaram-se, inicialmente, os dados de desempenho do IDS Snort na base de dados DARPA 1999 e, após isso, aplicou-se a técnica proposta para minimizar a TFP. O autor não deixou claro o quanto a TFP foi minimizada. Por outro lado, ele informou que, no caso de vulnerabilidades críticas, a acurácia do detector aumentou de 83,24% para 97,85%, sem alterar a abrangência, que se manteve em 37,47%. Já para o caso de vulnerabilidades não críticas, a acurácia aumentou de 71,31% para 95,56%, com uma pequena degradação na abrangência (de 35,67% para 32,43%). Assim, uma vez que a acurácia melhorou sem alterar significativamente a abrangência, pode-se dizer que houve uma diminuição na TFP.

Outro artigo que também utilizou o Snort para estudar o problema de falso positivo foi Tjhai *et al.* [49]. Neste caso, o IDS foi instalado para monitorar o tráfego de entrada e saída no servidor de páginas da Universidade de Plymouth (Reino Unido). Um dos

problemas dessa abordagem é que somente os eventos que geravam alarmes eram analisados. Isso significa que não houve conhecimento algum a respeito da quantidade de verdadeiros ou falsos negativos. Dessa forma, convencionou-se chamar de Taxa de Verdadeiro Positivo (TVP) a proporção de alarmes verdadeiros em relação à quantidade total de alarmes. Na verdade, essa proporção denota a precisão do IDS, e não a TVP. Analogamente, convencionou-se chamar de TFP a proporção de falsos alarmes em relação à quantidade total de alarmes, o que, na verdade, é o complemento da precisão. A despeito disso, os dados apresentados foram suficientes para mostrar a extensão do problema de falsos positivos, uma vez que estes foram consideravelmente mais numerosos em relação aos alarmes verdadeiros, nesse caso, em uma proporção aproximada de 24:1. Outro aspecto apresentado foi uma terceira categoria de alarmes (além de verdadeiro e falso positivo), denominada de irrelevante positivo, que, neste artigo (e em Subba, Biswas e Karmakar [48]), foi incluída na categoria de falso positivo. Um irrelevante positivo é um ataque sabidamente sem sucesso, pois explora vulnerabilidades inexistentes na rede em questão. Por exemplo, um ataque que explora uma determinada vulnerabilidade específica do SO Windows é um irrelevante positivo, caso todas as máquinas conectadas à rede rodem somente o SO Linux. Em resumo, percebeu-se que a grande maioria de falsos alarmes era gerada por três tipos de assinaturas. Dessa forma, por meio da análise de um especialista, foram feitos ajustes nas regras relacionadas a essas assinaturas, resultando na diminuição da proporção de falsos positivos (em relação a verdadeiros positivos) de 95,5% para 86,8%.

Zohrevand e Glässer [50] analisaram vários trabalhos que têm por objetivo diminuir a TFP de IDSs baseados em anomalia para níveis aceitáveis. De acordo com os autores, uma considerável atenção tem sido dada somente à obtenção de um modelo treinado em detecção de anomalias, sendo negligenciados aspectos relacionados à análise das decisões tomadas por esses modelos, especialmente as análises voltadas para mitigação de alarmes falsos, como, por exemplo, a "pontuação da anomalia", que mede o nível de diferença em relação a eventos normais, e ajuste do limiar de decisão. Dessa forma, diferentes formas de computar a pontuação da anomalia foram apresentadas, abordando probabilidades, correlações e similaridades entre anomalias. Infelizmente, a pesquisa não apresentou dados numéricos para fins de comparação.

6.3. Trabalhos com foco na explicabilidade do modelo

Nesta seção, serão apresentados trabalhos relacionados à explicabilidade de algoritmos aplicados em detecção de ciberataque. Normalmente, nesse caso, existem duas abordagens mais comuns: uma análise ampla dos atributos, onde são mostrados, de maneira geral, quais os mais importantes, e uma análise mais específica, onde, para uma determinada classificação (ou grupo de classificações), verificam-se quais os atributos que mais a influenciaram. Além disso, dados de desempenho já não são tão relevantes como nos trabalhos citados na subseção 6.1. Assim, o foco de trabalhos com explicabilidade está mais voltado à interpretação dos atributos e o quanto é razoável a importância dada a eles pelo algoritmo em uma dada tarefa de classificação. Marino, Wickramasinghe e Manic [30] analisaram exemplos classificados de forma errônea pelo modelo através de um método denominado de Aproximação Adversária. A ideia dessa técnica é verificar qual a menor mudança possível nos atributos necessária para corrigir uma classificação errada. Esse método funciona para qualquer tipo de modelo (linear, rede neural, SVM etc.) desde que a função de perda – normalmente, a entropia cruzada para o caso de classificação – possua um gradiente definido em relação aos atributos de entrada. Note que um *hacker* pode usar esse mesmo método para fazer o oposto, ou seja, determinar qual é a menor mudança nos atributos de um ataque (que foi detectado) necessária para que o modelo passe a considerá-lo como um tráfego legítimo. Entretanto, existem correlações entre os atributos do ataque que impõem restrições capazes de impedir alterações de forma independente. Dessa forma, a menor alteração teórica pode resultar num ataque inviável, visto que

a técnica de Aproximação Adversária desconsidera relações de interdependência entre atributos.

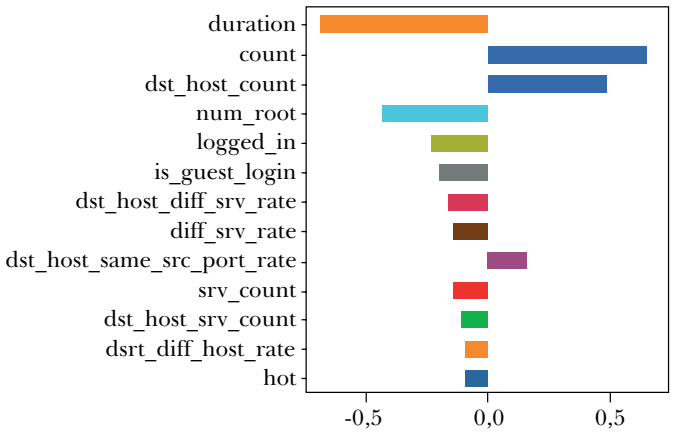

Fig. 4 – Amostras normais classificadas erroneamente como DoS. **Fonte: [30].**

A **Figura 4** elucida uma aplicação de Aproximação Adversária referente a um caso particular da base de dados NSL-KDD. Trata-se das amostras de tráfego normal (legítimo) classificadas erroneamente como DoS por um modelo de rede neural já treinado, com acurácia de 95,5%. No eixo vertical, constam os atributos, e, no eixo horizontal, os valores mínimos, na média, que deveriam ser subtraídos desses atributos para que o modelo passasse a classificar corretamente essas amostras. Isso significa que, ao realizar tais alterações mínimas nos atributos, amostras erroneamente classificadas como DoS passariam a ser classificadas corretamente em tráfego normal. Como resultado, é possível explicar a classificação errada. Note que ocorreu relativamente um alto número de conexões para o mesmo *host* (*count*) e para o mesmo endereço de destino (*dst_host_count*), sendo que essas conexões tiveram baixa duração (*duration*). Além disso, houve poucas operações realizadas como *root* nessas conexões (*num_root*), bem como baixo percentual de amostras que conseguiram se logar com sucesso (*logged_in*, *is_guest_login*). Essas características são comuns em ataques de negação de serviço, daí a razão para a classificação errônea. Caberia ao analista verificar se houve um erro de rótulo na base, ou se ocorreu apenas uma coincidência, ou, ainda, se há alguma razão lógica para registros rotulados como normais possuírem tais atributos.

Reyes *et al.* [51] utilizaram AM na detecção de intrusão em redes Wi-Fi, constantes na base AWID2. A detecção ocorre em dois estágios: o primeiro, por intermédio do RF, classificando os eventos em normal, inundação e personificação/injeção; e o segundo estágio separa a personificação da injeção, via NB. Esse modelo foi concebido como consequência de um trabalho anterior, onde os autores, ao classificarem os ataques num único estágio, perceberam que muitas amostras da classe personificação eram classificadas como injeção e vice-versa. Levando em consideração apenas a tarefa de detecção do primeiro estágio, ou seja, classificar em normal ou ataque, o modelo proposto obteve uma acurácia de 99,41%, uma abrangência de 94,1% e uma TFP de 0,13%. Adicionalmente, os autores dedicaram uma seção para a análise dos atributos utilizando a biblioteca SHAP (*SHapley Additive exPlanations*) [52], que oferece uma variedade de ferramentas para análise de explicabilidade global e local do modelo, inclusive de forma gráfica. Entretanto, esse trabalho não apresenta uma explicação de domínio, ou seja, qual o significado de cada atributo (ou, pelo menos, dos principais) e se é coerente ou não o comportamento do modelo, esclarecido pelas técnicas de explicabilidade. A despeito disso, os autores limitaram-se a citar quais atributos impactam positivamente ou negativamente na classificação de determinado rótulo. Entretanto, seria necessária uma análise mais profunda sobre a coerência desses impactos, o que só é possível com um conhecimento sobre o significado dos atributos envolvidos.

Wang *et al.* [53] também utilizaram SHAP para explicar as decisões de um IDS, além de reivindicar pioneirismo na aplicação dessa técnica de explicabilidade em detecção de intrusão. A base NSL-KDD foi utilizada para treinar dois modelos de rede neurais distintos: um contra todos e multiclasse, os quais obtiveram, respectivamente: acurácia de 80,6%, abrangência de 80,6% e TFP de 19,4%; acurácia de 80,3%, abrangência de 80,3% e TFP de 19,7%. Nesses modelos, foi feita uma análise de explicabilidade local referente aos ataques

DoS do tipo Neptune, que tenta saturar um servidor enviando uma elevada quantidade de pacotes SYN em todas as portas. Esses ataques possuem, portanto, uma alta taxa de conexões com erro SYN e, utilizando análise local SHAP, verificou-se que os modelos um contra todos e multiclasse classificaram os ataques Neptune em DoS com, em média, 93% e 89% de certeza, respectivamente. Entretanto, os atributos que direcionaram tal classificação foram bem distintos. De fato, o modelo um contra todos utilizou atributos mais razoáveis e diretamente relacionados ao Neptune. Tais atributos indicavam altas taxas de erro de conexão SYN, demonstrando que tal característica foi a principal responsável pela classificação. Já o modelo multiclasse não utilizou, de forma predominante, atributos relacionados a esse aspecto de erro SYN. Por isso, os autores concluíram que o modelo multiclasse não oferece razões consistentes para a confiança de um especialista nos resultados. Por outro lado, não houve uma análise crítica sobre os atributos que direcionaram a classificação feita pelo modelo multiclasse, ainda que eles não estivessem diretamente relacionados ao Neptune. Foi apresentada também uma análise global da importância dos atributos utilizando SHAP. A **Figura 5** mostra os 20 dentre os 41 atributos mais importantes para a classificação em DoS para o caso do modelo um contra todos. Cada linha na figura contém um atributo e todos os exemplos do conjunto teste na forma de pontos, cuja cor varia do azul ao vermelho, representando os menores e maiores valores do atributo, respectivamente. A localização dos pontos indicada no eixo horizontal denota quanto o respectivo atributo contribuiu a favor ou contra a classificação em DoS.

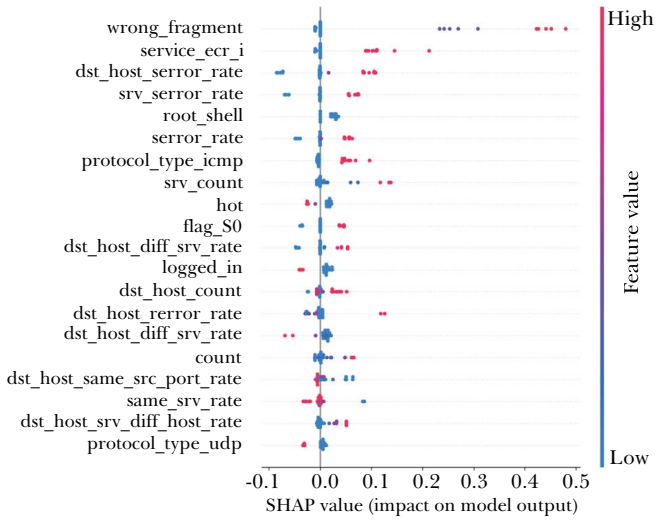

Fig. 5 – 20 atributos mais relevantes na classificação em DoS pelo modelo um contra todos. **Fonte: [53].**

A **Tabela 2** relaciona os quatro atributos mais relevantes da **Figura 5** com ataques do tipo DoS, demonstrando que, realmente, tais atributos são bastante pertinentes para uma efetiva classificação em DoS.

Tab. 2 – Tipos de ataque DoS.

Tipo	Descrição	Atributo
Land	Esse ataque trava o SunOS 4.1 pelo envio de um pacote TCP SYN mascarado, com o endereço de origem igual ao de destino.	*srv_serror_rate* *dst_host_serror_rate*
Neptune	Também conhecido por SYN *flood* ou ataque *half open*, ele inunda servidores web com pacotes TCP SYN mascarados, esgotando a memória. Consequentemente, novas conexões serão rejeitadas até o término da validade do tempo dessas conexões abertas pelos pacotes mascarados.	
PoD	Conhecido por "Ping da Morte" (do inglês *Ping of Death*), esse ataque envia pacotes IPs fragmentados, de maneira que, ao serem reconstruídos no *host* de destino, resultem em um pacote IP com tamanho acima de 65.535 bytes, que é o máximo permitido. Isso causa o travamento de sistemas operacionais antigos.	*wrong_fragment*
Teardrop	Esse ataque envia pacotes IPs fragmentados impossíveis de serem reconstruídos no *host* de destino, por haver sobreposições entre os fragmentos, causando o travamento de alguns sistemas operacionais.	
Smurf	Esse ataque envia pacotes ICMP do tipo *Echo Request*, conhecido como ping, para todos os *hosts* presente na rede da vítima, via *broadcast*. Como o campo endereço de origem desses pacotes foi mascarado com o endereço IP da vítima, esta recebe uma quantidade excessiva de pacotes *Echo Reply*, vindos como resposta de todos os *hosts* contidos no endereço de *broadcast*.	*service_ecr_i*

Fonte: Adaptado de [53].

Uma abordagem ligeiramente diferente foi apresentada em Mahdavifar e Ghorbani [54]. Nesse artigo, os autores inicialmente treinaram uma rede neural DNN para detectar ciberataques e, posteriormente, derivaram dessa rede um modelo especialista, denominado DeNNeS (do inglês *Deep Embedded Neural Network Expert System*). Modelos especialistas são definidos como um programa de computador que desempenha uma tarefa normalmente realizada por uma pessoa especialista por meio de regras se-então [55]. Nesse caso, essas regras foram extraídas do modelo DNN, que é não interpretável,

para compor a base de conhecimento de um sistema especialista, que é interpretável. Para uma definição formal dessas regras, considere T: $\{X, \mathbf{y}\} \rightarrow \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(m)}, y^{(m)})\}$ uma base de dados com m exemplos $(\mathbf{x}^{(i)}; y^{(i)})$ onde $\mathbf{x}^{(i)}$ representa um vetor de atributos do exemplo i, e $y^{(i)}$ o rótulo da classe associada. $\mathbf{x}^{(i)}$ possui n atributos $x_k^{(i)}$, como se segue: $\mathbf{x}^{(i)} = \{x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\}$; e $y^{(i)}$ é um número inteiro: $y^{(i)} \in Z$.

Assim, uma regra de classificação r_i, com respeito a $(\mathbf{x}^{(i)};y^{(i)})$, é definida como r_i: $P_i \rightarrow Q_i$, onde o antecedente da regra P_i é uma combinação de $l < n$ valores de atributos $\{k_1, k_2, \ldots, k_l\}$:

$$P_i = \{x_{k_1}^{(i)} \wedge x_{k_2}^{(i)} \wedge \ldots \wedge x_{k_l}^{(i)}\},$$

E o consequente da regra $Q_i = \{y^{(i)}\}$ é o rótulo da classe.

Dessa forma, Mahdavifar e Ghorbani [54] desenvolveram extensões do algoritmo MIJ (MACIE's *Inference Justification*) [55], com o objetivo de extrair essas regras do modelo DNN. Diferentes regras aplicadas a uma mesma amostra podem resultar em classificações diferentes. Assim, para realizar a classificação final baseada em regras, estabeleceu-se um sistema de votação, levando em conta todas as regras aplicáveis à amostra em questão. Além disso, o peso do voto de cada regra varia de acordo com a força da regra (η_{ri}), medida pelo produto de dois fatores: confiança (Cf_{ri}) e abrangência (Cr_{ri}), algo de certa forma análogo aos indicadores de desempenho precisão e abrangência de modelos classificadores. O primeiro fator é igual à proporção de exemplos que obedece à regra por completo em relação a todos que obedecem ao antecedente da regra. O segundo é igual à proporção de exemplos que obedece à regra por completo em relação a todos que obedecem ao consequente da regra. De maneira formal, tem-se:

$$\eta_{r_i} = Cf_{r_i} \cdot Cr_{r_i},$$

$$\text{onde} \quad Cf_{r_i} = \frac{|\{(x^{(i)}, y^{(i)}) \in T \mid P_i \subseteq x^{(i)} e Q = \{y^{(i)}\}\}|}{|\{(x^{(i)}, y^{(i)}) \in T \mid P_i \subseteq \mathbf{x}^{(i)}\}|}$$

$$\text{e} \quad Cr_{r_i} = \frac{|\{(x^{(i)}, y^{(i)}) \in T \mid P_i \subseteq x^{(i)} e Q_i = \{y^{(i)}\}\}|}{|\{(x^{(i)}, y^{(i)}) \in T \mid Q_i = \{y^{(i)}\}\}|}.$$

Foram utilizadas duas bases de dados para a aplicação dessa técnica: Phishing Websites, da Universidade da Califórnia Irvine (UCI), e uma base de malwares de Android, coletada por meio dos sites Virustotal e Contagio Security Blog. Nessas bases, foram aplicados DNN, DeNNeS e algoritmos clássicos de AM. DeNNeS obteve o segundo melhor desempenho, ficando bastante próximo do modelo DNN, que obteve o melhor. Na base Phishing Websites, DeNNeS obteve uma acurácia de 97,5% e uma TFP de 1,8%, superando uma outra técnica de extração de regras conhecida por JRip, além de superar também outras técnicas de AM (RF, SVM, KNN e NB). Já na base de malware, DeNNeS alcançou uma acurácia de 95,8% e uma TFP de 8%; da mesma forma, superou o JRip e as demais técnicas de AM.

6.4. Comparação entre os trabalhos relacionados

A **Tabela 3** apresenta os dados de desempenho dos trabalhos constantes na subseção 6.1. Como alguns trabalhos da subseção 6.3 também disponibilizaram dados de desempenho [51, 53, 54], eles foram incluídos ao final dessa tabela. O objetivo dessa comparação de resultados é mostrar que, em geral, a TFP não é desprezível, podendo tornar-se um problema importante em casos de tráfego elevado e desbalanceado em favor de eventos legítimos. Além disso, há uma variação muito grande desses valores, inclusive em trabalhos que utilizaram a mesma base de dados. Como praticamente todos os artigos abordados não disponibilizam em detalhe a metodologia e todos os dados necessários para a replicação do experimento, não há como garantir uma comparação completamente justa.

Tab. 3 – Desempenho obtido em artigos de detecção de intrusão utilizando AM.

Artigo	Base	Técnica	Abrangência (%)	TFP (%)	Acurácia (%)
Fan *et al.* [32]	DARPA 1998	RIPPER	94	2	-
Hu, Liao e Vemuri [33]	DARPA 1998	RSVM	100	3	-
Bivens *et al.*[34]	DARPA 1999	SOM/MLP	-	76	-
Kruegel *et al.* [35]	DARPA 1999	Redes Bayesianas	100	0,2	-

(continua...)

Tab. 3 - Continuação

Artigo	Base	Técnica	Abrangência (%)	TFP (%)	Acurácia (%)
Shon e Moon [36]	DARPA 1999	SVM aprimorada	72,73	10,2	-
Tajbakhsh, Rahmati e Mirzaei [37]	KDD 1999	Regras Fuzzy	100	13	-
Amor, Benferhat e Elouedi [38]	KDD 1999	NB	89	2	-
Chen, Hsu e Shen [39]	DARPA 1998	SVM	100	8,53	-
Adebowale, Idowu e Amarachi [40]	NSL-KDD	SVM	95,9	1,4	97,3
		MLP	95,9	4,4	95,8
		NB	87,7	8,8	89,6
		DT	99,6	0,4	99,6
Thaseen e Kumar [41]	NSL-KDD	NBTree	97,8	4,8	-
Ustebay, Turgut e Aydin [42]	CIC-IDS 2017	MLP	100	18	-
Le, Kim e Kim [43]	KDD 1999	LSTM	98,95	9,98	-
Xu *et al*. [44]	KDD 1999	GRU/MLP	99,42	0,05	-
	NSL-KDD		99,31	0,84	-
Papamartzivanos [45]	KDD 1999	GA/DT	98,24	0,75	98,85
	NSL-KDD		95,97	1,08	97,55
	UNSW-NB15		63,76	2,61	84,33
Stefanova [47]	NSL-KDD	RF	99,9	0,16	-
Reyes *et al*. [51]	AWID	RF	94,1	0,13	99,41
Wang *et al*. [53]	NSL-KDD	MLP	80,6	19,4	80,6
Mahdavifar e Ghorbani [54]	Privada	DNN	-	1,8	97,5

Fonte: Elaborado pelos autores.

Ainda na **Tabela 3**, verifica-se que alguns trabalhos obtiveram uma TFP menor que 1%, com uma abrangência de 99%. Esse elevado desempenho ocorreu uma vez na base DARPA 1999, uma vez na KDD 1999 e três vezes na NSL-KDD. Os algoritmos utilizados foram Redes Bayesianas, DT, RF e uma combinação de GRU com MLP. Sem fazer uma análise de explicabilidade nesses modelos, não há como se ter uma ideia se haverá uma manutenção de tais desempenhos em aplicações reais. Embora tenha sido de alta relevância o desenvolvimento dos métodos e os desempenhos alcançados, também é necessário entender se os atributos utilizados para detecção representam relações gerais que não estão restritas às bases de dados empregadas.

Ainda que tais desempenhos se mantenham em aplicações reais, mesmo a menor TFP obtida, no caso, 0,05% para a base KDD 1999, pode gerar uma quantidade considerável de falsos positivos em redes altamente velozes e desbalanceadas, ou seja, com uma quantidade de tráfegos benignos muito superior àquelas de tráfegos maliciosos. Lembrando ainda que a base KDD 1999, além de ser antiga, possui imperfeições que facilitam a tarefa de classificação do modelo [19]. Além disso, o artigo que a empregou [44] é bem mais recente que a data de construção dela. Isso tudo corrobora para um desempenho superestimado. Por outro lado, note que a abrangência de 100% obtida na base CIC-IDS 2017 só foi possível com uma TFP de 18%, bem acima de 0,05%. Já na base UNSW-NB15, disponibilizada em 2015, a TFP foi de 2,61%, porém, com uma baixa abrangência (63,76%). Certamente, um ajuste no limiar do detector para aumentar esta abrangência ocasionaria um considerável aumento na TFP. Entretanto, é necessário ampliar a pesquisa para obter mais dados a respeito de desempenhos obtidos em bases mais recentes.

A **Tabela 4** sumariza os trabalhos que apresentaram outras propostas, além de somente obterem alto índice de desempenho. À exceção de [45], constante na subseção 6.1, todos os trabalhos nessa tabela foram extraídos da subseção 6.2 e da subseção 6.3. Portanto, em geral, as propostas desses trabalhos relacionam-se a estratégias para tratar a ocorrência de falsos positivos, ou, então, a aspectos de explicabilidade, no intuito de obter conhecimento sobre quais combinações de valores de atributos representam de fato um ataque, e se há coerência nessa combinação.

Tab. 4 – Artigos sobre detecção de intrusão com outros objetivos além do desempenho.

Artigo	Bases	Técnica	Objetivo	Resultado
Papamartzivanos [45]	KDD 1999 e NSL-KDD	STL	Desenvolver um IDS baseado em assinatura autoadaptável.	IDS autoadaptável superou IDS estático em ambientes diversos.

(continua...)

Tab. 4 - Continuação

Artigo	Bases	Técnica	Objetivo	Resultado
Subba, Biswas e Karmakar [48]	DARPA 1999	Filtragem de detecção baseada em vulnerabilidades.	Redução da TFP em IDS de assinatura.	Aumento da acurácia, sem considerável degradação na abrangência.
Tjhai *et al*. [49]	Privada	Ajustes nas regras do IDS relacionadas às assinaturas que mais geram falsos positivos.	Redução da TFP em IDS de assinatura.	Diminuição na proporção de falsos positivos (em relação à quantidade total de alarmes) de 95,5% para 86,8%.
Marino, Wickramasinghe e Manic [30]	NSL-KDD	Aproximação Adversária	Entender o motivo da ocorrência de falso positivo em um grupo de exemplos, de tráfego normal, classificados em ataque DoS.	Obtenção dos atributos que mais contribuíram para a classificação errônea, bem como o quanto tais atributos deveriam ser alterados para uma correta classificação.
Wang *et al*. [53]	NSL-KDD	SHAP	Verificar quais atributos foram mais usados pelo modelo na classificação de ataques Neptune em DoS, e se há coerência nessa escolha de atributo.	Obtenção dos atributos que mais contribuíram para a classificação do Netpune em DoS. Houve mais coerência no modelo MLP um contra todos do que no MLP multiclasse.
Mahdavifar e Ghorbani [54]	Privada	Extração de regras do modelo	Substituir um modelo DNN não interpretável, por outro interpretável.	Obtenção de um modelo especialista, por meio de extração de regras do modelo DNN, com pouca degradação no desempenho.

Fonte: Elaborado pelos autores.

7. Conclusão

AM tem se mostrado um campo de pesquisa promissor na atividade de detecção de intrusão, em virtude do elevado desempenho obtido em bases de dados simuladas. Entretanto, não há garantias de que o mesmo nível de desempenho seja obtido em aplicações reais. São necessários, portanto, estudos que verifiquem o nível de semelhança entre bases simuladas e tráfegos reais. Um obstáculo que se apresenta é a possibilidade de exposição de informações sensíveis contidas nesse tipo de tráfego ao torná-lo público para estudos.

Outra abordagem para analisar possibilidades de manutenção do desempenho em aplicações reais seria por meio da explicabilidade. Essa técnica permite verificar os fatores considerados mais importantes por modelos de AM complexos na tarefa de detectar intrusão. Assim, o estudo desses fatores pode concluir se eles realmente representam características gerais, relacionadas a ataques, ou tráfego benigno, independente do ambiente ser real ou simulado.

Finalmente, também foi verificado o problema da ocorrência de falso positivo, que pode levar à negligência de ataques verdadeiros, camuflados por uma quantidade excessiva de alarmes. IDSs que utilizam AM são mais suscetíveis a essas circunstâncias, o que, em geral, tem sido um obstáculo na implantação dessa tecnologia na detecção de intrusão.

Referências

[1] ALYASIRI, H. Developing Efficient and Effective Intrusion Detection System using Evolutionary Computation. Dissertação (PhD em Ciência Computacional) – University of York, Heslington, 2018.

[2] AKSU D.; AYDIN M. A. Detecting Port Scan Attempts with Comparative Analysis of Deep Learning and Support Vector Machine Algorithms. In International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT). Piscataway: IEEE, 2018. p. 77–80. http://dx.doi.org/10.1109/IBIGDELFT.2018.8625370.

[3] HACKER tries to poison water supply of Florida city. BBC News, 8 fev. 2021. Disponível em: https://www.bbc.com/news/world-us-canada-55989843. Acesso em: fev. 2021.

[4] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION – ISO; INTERNATIONAL ELECTROTECHNICAL COMMISSION – IEC. Joint Technical Committee ISO/IEC JTC 1/SC 127. ISO/IEC 27032:2012(en) Information technology — Security techniques — Guidelines for cybersecurity. Whashington, DC: ISO: IEC, 2012. Disponível em: https://www.iso.org/obp/ui/#iso:std:iso-iec:27032:ed-1:v1:en. Acesso em: 24 set. 2021.

[5] AMOROSO, E. G.; AMOROSO, M. E. From CIA to APT: An Introduction to Cyber Security. 2017.
[6] SCARFONE, K. A.; MELL, P. M. NIST Special Publication 800-94 – Guide to Intrusion Detection and Prevention Systems (IDPS): Recommendations of the National Institute of Standards and Technology. Gaithersburg, MD: National Institute of Standards and Technology, 2007.
[7] INDIAN CYBER SECURITY SOLUTIONS. Intrusion Detection System and its Detailed Working Function. ICSS, 2021. Disponível em: https://indiancybersecuritysolutions.com/intrusion-detection-system-working-function. Acesso em: 17 ago. 2021.
[8] THOMA, M. Receiver Operating Characteristic (ROC) curve with False Positive Rate and True Positive Rate. Wikimedia Commons, 2018. Disponível em: https://commons.wikimedia.org/w/index.php?title=File:Roc-draft-xkcd-style.svg&oldid=491003296. Acesso em: 7 jan. 2021.
[9] XIN, Y.; KONG, L.; LIU, Z.; CHEN, Y.; LI, Y.; ZHU, H. et al. Machine Learning and Deep Learning Methods for Cybersecurity. IEEE Access, v. 6, p. 35365–35381, 2018. https://doi.org/10.1109/ACCESS.2018.2836950.
[10] KELLEHER, J. D.; NAMEE B. M.; D'ARCY, A. Fundamentals of Machine Learning for Predictive Data Analytics. 2. ed. Cambridge: MIT Press, 2020.
[11] MUELLER, J. P.; MASSARON, L. Machine Learning for Dummies. Hoboken: John Wiley & Sons, 2016.
[12] GHAHRAMANI, Z. Unsupervised Learning. In BOUSQUET, O.; VON LUXBURG, U.; RÄTSCH, G. (ed.). Advanced Lectures on Machine Learning. ML Summer Schools 2003. Amsterdam: Springer, 2003. https://doi.org/10.1007/978-3-540-28650-9_5.
[13] EVSUKOFF, A. G. Inteligência Computacional: Fundamentos e Aplicações. Rio de Janeiro: E-papers, 2020.
[14] VANDERPLAS, J. Python Data Science Handbook: Essential Tools for Working with Data. Sebastopol: O'Reilly, 2016.
[15] RASCHKA S.; MIRJALILI, V. Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow. 2. ed. Birmingham: Packt Publishing, 2017.
[16] BUCZAK, A. L.; GUVEN, E. A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. IEEE Communications Surveys & Tutorials, v. 18, n. 2, p. 1153–1176, 2016. https://doi.org/10.1109/COMST.2015.2494502.
[17] DARPA Intrusion Detection Evaluation Dataset. MIT Lincoln Laboratory, 1998/1999. Disponível em: https://www.ll.mit.edu/r-d/datasets. Acesso em: 27 ago. 2021.
[18] DATA – KDD Cup 1999: Computer network intrusion detection. SIGKDD, 1999. Disponível em: https://www.kdd.org/kdd-cup/view/kdd-cup-1999/Data. Acesso em: 27 ago. 2021.
[19] TAVALLAEE, M.; BAGHERI, E.; LU, W.; GHORBANI, A. A. A detailed analysis of the KDD CUP 99 data set. In IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA). Piscataway: IEEE, 2009. p. 1–6. https://doi.org/10.1109/CISDA.2009.5356528.
[20] NSL-KDD DATASET. Canadian Institute for Cybersecurity. University of New Brunswick, 2009. Disponível em: https://www.unb.ca/cic/datasets/nsl.html. Acesso em: 27 ago. 2021.
[21] CTU-13 DATASET. A Labeled Dataset with Botnet, Normal and Background Traffic. Stratosphere Lab, 2011. Disponível em: https://www.stratosphereips.org/datasets-ctu13. Acesso em: 29 ago. 2021.
[22] DELPLACE, A.; HERMOSO S.; ANANDITA, K. Cyber Attack Detection thanks to Machine Learning Algorithms. arXiv preprint, 2001.06309, 2020.
[23] DATASETS. Canadian Institute for Cibersecurity, University of New Brunswick, 2020. Disponível em: https://www.unb.ca/cic/datasets/index.html. Acesso em: 29 ago. 2022.
[24] WIRELESS DATASETS. University of the Aegean, 2021. Disponível em: https://icsdweb.aegean.gr/awid/download-dataset. Acesso em: 23 dez. 2021.
[25] KOLIAS, C.; KAMBOURAKIS, G.; STAVROU, A.; GRITZALIS, S. Intrusion Detection in 802.11 Networks: Empirical Evaluation of Threats and a Public Dataset. IEEE Communications Surveys & Tutorials, v. 18, n. 1, p. 184–208, 2016. https://doi.org/10.1109/COMST.2015.2402161.
[26] MOUSTAFA, N.; SLAY, J. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Military Communications and Information Systems Conference (MilCIS). Piscataway: IEEE, 2015. p. 1–6. https://doi.org/10.1109/MilCIS.2015.7348942.
[27] GARCIA, S.; PARMISANO, A.; ERQUIAGA, M. J. IoT-23: A labeled dataset with malicious and benign IoT network traffic (Version 1.0.0). Zenodo. Stratosphere Lab, 2020. Disponível em: https://www.stratosphereips.org/datasets-iot23. Acesso em: 23 dez. 2021.

[28] USB-IDS-1. Università degli Studi del Sannio di Benevento, 2021. Disponível em: http://idsdata.ding.unisannio.it/datasets.html. Acesso em: 25 dez. 2021.
[29] NISIOTI, A.; MYLONAS, A.; YOO, P. D.; KATOS, V. From Intrusion Detection to Attacker Attribution: A Comprehensive Survey of Unsupervised Methods. IEEE Communications Surveys & Tutorials, v. 20, n. 4, p. 3369–3388, 2018. https://doi.org/10.1109/COMST.2018.2854724.
[30] MARINO, D. L.; WICKRAMASINGHE, C. S.; MANIC, M. An Adversarial Approach for Explainable AI in Intrusion Detection Systems. In 44th Annual Conference of the IEEE Industrial Electronics Society. Piscataway: IEEE, 2018. p. 3237–3243. https://doi.org/10.1109/IECON.2018.8591457.
[31] KHRAISAT, A.; GONDAL, I.; VAMPLEW, P.; KAMRUZZAMAN, J. Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity, v. 2, n. 20, 2019. https://doi.org/10.1186/s42400-019-0038-7.
[32] FAN, W.; MILLER, M.; STOLFO, S. J.; LEE W.; CHAN, P. K. Using artificial anomalies to detect unknown and known network intrusions. In IEEE International Conference on Data Mining. Piscataway: IEEE, 2001. p. 123–130. https://doi.org/10.1109/ICDM.2001.989509.
[33] HU, W.; LIAO, Y.; VEMURI, V. R. Robust Support Vector Machines for Anomaly Detection in Computer Security. In International Conference on Machine Learning and Applications (ICMLA). Piscataway: IEEE, 2003. p. 168–174.
[34] BIVENS, A.; PALAGIRI, C.; SMITH, R.; SZYMANSKI, B.; EMBRECHTS, M. Network-Based Intrusion Detection Using Neural Networks. In Intelligent Engineering Systems through Artificial Neural Networks ANNIE-2002, v. 12. New York: ASME Press, 2002. p. 579–584.
[35] KRUEGEL, C.; MUTZ, D.; ROBERTSON, W.; VALEUR, F. Bayesian event classification for intrusion detection. In 19th Annual Computer Security Applications Conference (ACSAC). Piscataway: IEEE, 2003. p. 14–23. https://doi.org/10.1109/CSAC.2003.1254306.
[36] SHON, T.; MOON, J. A hybrid machine learning approach to network anomaly detection. Information Sciences, v. 177, n. 18, p. 3799–3821, 2007. https://doi.org/10.1016/j.ins.2007.03.025.
[37] TAJBAKHSH, A.; RAHMATI, M.; MIRZAEI, A. Intrusion detection using fuzzy association rules. Applied Soft Computing, v. 9, n. 2, p. 462–469, 2009. https://doi.org/10.1016/j.asoc.2008.06.001.
[38] AMOR, N. B.; BENFERHAT S.; ELOUEDI, Z. Naive Bayes vs decision trees in intrusion detection systems. In Proceedings of the 2004 ACM Symposium on Applied Computing, Association for Computing Machinery. New York: ACM Digital Library, 2004. p. 420–424. https://doi.org/10.1145/967900.967989.
[39] CHEN, W. H.; HSU, S. H.; SHEN, H. P. Application of SVM and ANN for intrusion detection. Computers & Operations Research, v. 32, n. 10, p. 2617–2634, 2005. https://doi.org/10.1016/j.cor.2004.03.019
[40] ADEBOWALE, A.; IDOWU, S. A.; AMARACHI, A. A. Comparative study of selected data mining algorithms used for intrusion detection. International Journal of Soft Computing and Engineering, v. 3, n. 3, p. 237–241, 2013.
[41] THASEEN, I. S.; KUMAR, C. A. An analysis of supervised tree based classifiers for intrusion detection system. In International Conference on Pattern Recognition, Informatics and Mobile Engineering. Piscataway: IEEE, 2013. p. 294–299. https://doi.org/10.1109/ICPRIME.2013.6496489.
[42] USTEBAY, S.; TURGUT, Z.; AYDIN, M. A. Intrusion Detection System with Recursive Feature Elimination by Using Random Forest and Deep Learning Classifier. In International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT). Piscataway: IEEE, 2018. p. 71–76. https://doi.org/10.1109/IBIGDELFT.2018.8625318.
[43] LE, T. T. H.; KIM, J.; KIM, H. An Effective Intrusion Detection Classifier Using Long Short-Term Memory with Gradient Descent Optimization. In International Conference on Platform Technology and Service (PlatCon). Piscataway: IEEE, 2017. p. 1–6. https://doi.org/10.1109/PlatCon.2017.7883684.
[44] XU, C.; SHEN, J.; DU, X.; ZHANG, F. An Intrusion Detection System Using a Deep Neural Network with Gated Recurrent Units. IEEE Access, v. 6, p. 48697–48707, 2018. https://doi.org/10.1109/ACCESS.2018.2867564.
[45] PAPAMARTZIVANOS, D. C. Advanced machine learning methods for network intrusion detection. Tese (Doutorado em Filosofia) – University of the Aegean, Mitilene, 2019.
[46] RAINA, R.; BATTLE, A.; LEE, H.; PACKER B.; NG, A. Y. Self-Taught Learning: Transfer Learning from Unlabeled Data. In Proceedings of the 24th International Conference on Machine Learning, Association for Computing Machinery. New York: ACM Digital Library, 2007. p. 759–766. https://doi.org/10.1145/1273496.1273592.
[47] STEFANOVA, Z. S. Machine Learning Methods for Network Intrusion Detection and Intrusion Prevention Systems. Tese (Doutorado em Filosofia) – University of South Florida, Tampa, 2018.

[48] SUBBA, B.; BISWAS, S.; KARMAKAR, S. False alarm reduction in signature-based IDS: game theory approach. Security and Communication Networks, v. 9, n. 18, p. 4863–4881, 2016. https://doi.org/10.1002/sec.1661.
[49] TJHAI, G. C.; PAPADAKI, M.; FURNELL, S. M.; CLARKE, N. L. Investigating the problem of IDS false alarms: An experimental study using Snort. In Proceedings of The Ifip Tc 11 23rd International Information Security Conference, v. 278. Laxenburg: IFIP, 2008. p. 253–267.
[50] ZOHREVAND, Z.; GLÄSSER, U. Should I Raise The Red Flag? A comprehensive survey of anomaly scoring methods toward mitigating false alarms. arXiv preprint, 1904.06646, 2019.
[51] REYES, A. A.; VACA, F. D.; AGUAYO, G. A. C.; NIYAZ, Q.; DEVABHAKTUNI, V. A Machine Learning Based Two-Stage Wi-Fi Network Intrusion Detection System. Electronics, v. 9, n. 10, p. 1689, 2020. https://doi.org/10.3390/electronics9101689.
[52] LUNDBERG, S. M.; LEE, S. I. A Unified Approach to Interpreting Model Predictions. In 31th Conference on Neural Information Processing Systems. New York: ACM Digital Library, 2017. p. 4768–4777.
[53] WANG, M.; ZHENG, K.; YANG, Y.; WANG, X. An Explainable Machine Learning Framework for Intrusion Detection Systems. IEEE Access, v. 8, p. 73127–73141, 2020. https://doi.org/10.1109/ACCESS.2020.2988359.
[54] MAHDAVIFAR, S.; GHORBANI, A. A. DeNNeS: deep embedded neural network expert system for detecting cyber attacks. Neural Computing and Applications, v. 32, n. 18, p. 14753–14780, 2020. https://doi.org/10.1007/s00521-020-04830-w.
[55] GALLANT, S. I. Neural Network Learning and Expert Systems. 3. ed. Cambridge: MIT Press, 1995.