

UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E CIÊNCIA DA COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

**Além da Detecção: Tornando os
Algoritmos de Detecção de Ataques
Cibernéticos mais Compreensíveis**

Thiago Duvanel Ferreira

MONOGRAFIA FINAL

MAC 499 — TRABALHO DE
FORMATURA SUPERVISIONADO

Supervisor: Prof. Dr. Daniel Macêdo Batista

São Paulo
2025

*O conteúdo deste trabalho é publicado sob a licença CC BY 4.0
(Creative Commons Attribution 4.0 International License)*

*Dedico esse trabalho ao meu pai, que serviu e ainda
serve de inspiração para tudo o que eu desejo ser na vida.*

Agradecimentos

*Que eu não fique nunca como esse velho inglês aí do lado,
que dorme numa cadeira à espera de visitas que não vêm.*

— Oswald de Andrade

Agradeço a toda minha família, parte fundamental de toda minha trajetória até o momento de criação desta tese: ao meu pai Celso, que é minha inspiração em praticamente todos os aspectos da minha vida, minha mãe Marcela que formou meu caráter, minhas avós Nismeria e Lourdes que me deram amor desde a infância e ao meu tio Antônio Wilson, que sempre foi meu parceiro. Agradeço aos meus amigos de infância, Felipe Lomeu, Felipe Garcia, Guilherme e Clarice que são e sempre serão minhas âncoras independente de onde estejam pelo mundo. Aos meus amigos de apartamento Flávio e Pedro que sempre foram um ombro amigo nos meus momentos mais difíceis. Agradeço aos meus amigos Hannah, Filipe, Guilherme e Laís, nos quais São Paulo me proporcionou e tornaram-se minha admiração para ter vontade de ser melhor todos os dias. Agradeço à minha equipe do Rabobank, que apostaram em mim desde o início para entrar na carreira dos meus sonhos e que resultou nesta tese. Agradeço imensamente à Maria Eduarda Hespanhol, que sabia o caminho certo para tomar da minha vida após o Ensino Médio, antes de mim mesmo. E, finalmente, agradeço ao meu orientador, Daniel Batista, que foi responsável pela minha entrada na área que pretendo seguir por toda minha vida.

Resumo

Thiago Duvanel Ferreira. **Além da Detecção: Tornando os Algoritmos de Detecção de Ataques Cibernéticos mais Compreensíveis**. Monografia (Bacharelado). Instituto de Matemática, Estatística e Ciência da Computação, Universidade de São Paulo, São Paulo, 2025.

Com o aumento da complexidade e da escala dos sistemas disponíveis em redes de computadores, modelos de aprendizado de máquina vêm sendo cada vez mais utilizados e desenvolvidos para detectar tráfego malicioso e prevenir vários tipos de ataques cibernéticos. Contudo, muitos desses modelos funcionam como "caixas opacas", sem oferecer informação suficiente sobre como suas decisões são tomadas. Essa falta de transparência compromete a confiança de administradores da rede, que precisam entender e justificar como um tráfego foi classificado como malicioso e as atitudes tomadas. Além disso, a falta de explicações pode levar à difícil manutenção do modelo e dificuldades em identificar e corrigir erros, desincentivando melhorias. Portanto, é fundamental que os modelos adotados para essa tarefa sejam explicáveis, permitindo que especialistas em redes — ainda que não tenham experiência em aprendizado de máquina — possam interpretar e avaliar as decisões de um sistema automatizado. Para tal finalidade, ferramentas que explicam as possíveis saídas de modelos de Inteligência Artificial possuem papel fundamental em contextos relacionados à detecção de intrusões. Este trabalho, então, investiga o impacto da aplicação de ferramentas de Inteligência Artificial Explicável, como o LIME (*Local Interpretable Model-agnostic Explanations*) e o SHAP (*SHapley Additive exPlanations*), na interpretação e validação de modelos de aprendizado de máquina utilizados na detecção de intrusões. O estudo avalia se a adoção de recomendações geradas por essas ferramentas, baseadas na importância das características que levaram a uma determinada classificação, resulta em uma melhoria mensurável no desempenho dos modelos, como a redução de falsos positivos e o aumento da precisão. Por meio de uma análise comparativa, busca-se demonstrar que a explicabilidade não é apenas um requisito de confiança, mas um fator crítico para otimizar a eficácia dos sistemas de segurança cibernética, ao permitir que especialistas atuem de forma mais precisa e informada para aprimorar os modelos e as estratégias de defesa. Por exemplo, foi possível perceber que o tamanho dos modelos finais diminuiu até em 22% comparado ao tamanho original, mantendo suas métricas de eficiência ótimas. Seguindo as boas práticas de ciência aberta todos os códigos desenvolvidos durante o desenvolvimento desse trabalho estão disponíveis como software livre.

Palavras-chave: Detecção de Intrusão. Redes de Computadores. Explicabilidade. Dataset. Aprendizado de Máquina.

Abstract

Thiago Duvanel Ferreira. **Beyond Detection: Making Cyberattack Detection Algorithms More Understandable**. Capstone Project Report (Bachelor). Institute of Mathematics, Statistics and Computer Science, University of São Paulo, São Paulo, 2025.

With the increasing complexity and scale of systems available on computer networks, machine learning models are being increasingly utilized and developed to detect malicious traffic and prevent various types of cyber-attacks. However, many of these models function as "opaque boxes," offering insufficient information about how their decisions are made. This lack of transparency undermines the trust of network administrators, who must understand and justify how traffic is classified as malicious and the resulting actions taken. Moreover, a lack of explanations can lead to difficult model maintenance, challenges in identifying and correcting errors, and disincentivize improvements. Therefore, it is fundamental that models adopted for this task be explainable, enabling network specialists—even those without machine learning experience—to interpret and evaluate the decisions of an automated system. To this end, tools that explain the outputs of Artificial Intelligence models play a pivotal role in contexts related to intrusion detection. This work, therefore, investigates the impact of applying Explainable Artificial Intelligence (XAI) tools, such as LIME (Local Interpretable Model-agnostic Explanations) and SHAP (SHapley Additive exPlanations), on the interpretation and validation of machine learning models used for intrusion detection. The study evaluates whether adopting recommendations generated by these tools, based on the feature importance that led to a given classification, results in a measurable improvement in model performance, such as reduced false positives and increased precision. Through a comparative analysis, this work seeks to demonstrate that explainability is not merely a requirement for trust, but a critical factor in optimizing the effectiveness of cybersecurity systems. It allows specialists to act more precisely and informedly to enhance both the models and the defense strategies. For instance, it was observed that the final model size could be reduced by up to 22% compared to the original size while maintaining optimal performance metrics, which facilitates implementation and reduces decision-making time. Furthermore, following open science best practices, all code developed during this work is available as open-source software.

Keywords: Intrusion Detection. Computer Networks. Explainability. Dataset. Machine Learning.

Lista de abreviaturas

LIME	<i>Local Interpretable Model-agnostic Explanations</i>
DFT	<i>Shapley Additive Explanations</i>
IDS	Sistema de Detecção de Intrusão (<i>Intrusion Detection System</i>)
XAI	Inteligência Artificial Explicável (<i>Explainable Artificial Intelligence</i>)
DoS	Negação de Serviço (<i>Denial of Service</i>)
DL	Aprendizado Profundo (<i>Deep Learning</i>)
IoT	Internet das Coisas (<i>Internet of Things</i>)

Lista de figuras

1.1	Ilustração de um ataque de Força Bruta.	10
1.2	Ilustração de um Ataque de Reconhecimento (Escaneamento de Portas). .	11
1.3	Ilustração de um Ataque de Negação de Serviço.	11
2.1	Exemplo de saída gerada pelo LIME, retirado de https://towardsdatascience.com	18
2.2	Exemplo de gráfico "espinha de peixe" gerado pelo SHAP, retirado de <i>SHAP (SHapley Additive exPlanations)</i> s.d.	19
3.1	Topologia de rede utilizada para criar o <i>dataset</i> , retirado de <i>DALAMAGKAS et al., 2025</i>	25
3.2	Comparação de peso das 15 primeiras features entre LIME e o SHAP do CICFlowMeter	35
3.3	Acurácia e F1-Score do <i>dataset</i> CICFlowMeter e suas modificações	36
3.4	Recall e Precisão do <i>dataset</i> CICFlowMeter e suas modificações	37
3.5	Duração do treinamento e tamanho dos modelos do <i>dataset</i> CICFlowMeter e suas modificações	38
3.6	Duração do treinamento e tamanho dos modelos do <i>dataset</i> OCPPFlowMeter e suas modificações	39
3.7	Acurácia e F1-Score do <i>dataset</i> OCPPFlowMeter e suas modificações . . .	40
3.8	Recall e Precisão do <i>dataset</i> OCPPFlowMeter e suas modificações	40
3.9	Comparação de peso das 15 primeiras features entre LIME e o SHAP do OCPPFlowMeter.	41

Lista de tabelas

2.1	Exemplo de análise de trade-off para o modelo otimizado. O F1-Score representa o equilíbrio entre detectar ameaças (<i>recall</i>) e evitar falsos alarmes (precisão).	20
3.1	Comparativo de importância de features para o CICFlowMeter (LIME vs. SHAP). As colunas LIME e SHAP representam a posição no ranking de importância nos respectivos métodos de explicabilidade.	32
3.2	Comparativo de importância de features para o OCPPFlowMeter (LIME vs. SHAP). As colunas LIME e SHAP representam a posição no ranking de importância nos respectivos métodos de explicabilidade.	33

Sumário

Introdução	1
Objetivos	3
1 Conceitos Básicos	5
1.1 Ataques em Redes de Computadores	5
1.2 Tratamento de Intrusão	7
1.3 Aprendizado de Máquina	8
1.4 Explicabilidade	11
1.4.1 LIME	12
1.4.2 SHAP	13
2 Arcabouço para Compreensão e Otimização de Modelos de Detecção de Intrusão	15
2.1 O Arcabouço	16
2.1.1 Estabelecendo o Modelo de Referência	16
2.1.2 Entendendo as Decisões do Modelo	17
2.1.3 Extração e Validação da Lógica	19
2.1.4 Otimização da Detecção	20
2.2 Recapitulação	21
3 Análise de Desempenho	23
3.1 Modelos Utilizados	23
3.1.1 OCPP-IDS	24
3.2 Resultados	30
3.2.1 OCPP-IDS	31
4 Conclusões	43

Introdução

A segurança cibernética vive uma era de transformação impulsionada pela Inteligência Artificial. Sistemas baseados em Aprendizado de Máquina (*Machine Learning*) e Aprendizagem Profunda (*Deep Learning*), especialmente em Sistemas de Detecção de Intrusão, tornaram-se essenciais por sua capacidade superior de analisar vastos volumes de dados e identificar ameaças complexas, incluindo ataques de dia-zero, com eficiência e precisão. Contudo, essa evolução tecnológica introduziu uma grande dúvida sobre a utilização de IA, pois quanto mais poderosos os modelos se tornam, menos transparentes eles ficam NEUPANE *et al.*, 2022.

Muitos desses sistemas de ponta operam como "caixas opacas" (*opaque boxes*), oferecendo resultados precisos sem, no entanto, fornecer justificativas compreensíveis para suas decisões. Essa opacidade é um ponto de vulnerabilidade crítico no domínio da segurança. Para um analista de redes, um alerta sem contexto é de valor limitado, dificultando a validação, a resposta a incidentes e, principalmente, a construção de confiança no sistema. Conforme aponta a literatura NEUPANE *et al.*, 2022, essa falta de transparência não apenas compromete a usabilidade, mas também dificulta a manutenção e a identificação de erros nos modelos, desincentivando sua melhoria contínua.

Para endereçar essa lacuna, emergiu o campo da Inteligência Artificial Explicável, que visa desenvolver métodos para que as decisões da IA sejam interpretáveis por humanos. A XAI não se resume a uma única técnica, mas a um conjunto de princípios que incluem transparência, confiança, usabilidade e causalidade. A pesquisa na área mostra que a maioria das abordagens de XAI aplicadas à cibersegurança são do tipo *post-hoc* — ou seja,

buscam explicar o modelo depois de treinado [CAPUANO et al., 2022](#). Mais importante, essas explicações precisam ser direcionadas ao usuário final, como o especialista em segurança, que necessita de especificações claras e não apenas de uma análise abstrata, que pode não ser reconhecível por ele, principalmente se não tiver conhecimento em aplicações da Inteligência Artificial.

Entretanto, a busca pela transparência pode ser muito complexa. A mesma informação que torna um modelo compreensível para um defensor pode ser explorada por um atacante para encontrar suas fraquezas [CAPUANO et al., 2022](#). A explicabilidade pode expor o modelo a riscos como *ataques adversariais* (*adversarial attacks*), nos quais um invasor insere pequenas perturbações nos dados de entrada para enganar o classificador, ou *ataques de envenenamento* (*poisoning attacks*), que manipulam os dados de treinamento para comprometer o comportamento do modelo. Assim, a explicabilidade assume duas faces ao ser essencial para a confiança e a operação, mas também representa um potencial vetor de ataque contra a própria IA.

É neste cenário que o presente trabalho se posiciona. A pesquisa investiga a aplicação de ferramentas de XAI — especificamente o LIME e o SHAP [NEUPANE et al., 2022](#) — não somente como um meio para gerar confiança e interpretabilidade para o analista de segurança de redes, mas também como um mecanismo para otimizar e, potencialmente, fortalecer o próprio modelo de detecção. A hipótese central é que, ao utilizar as informações da importância das características fornecidas pela XAI, é possível redesenhar o modelo para torná-lo mais enxuto, eficiente e focado nos indicadores de ameaça mais críticos. Este processo de simplificação pode, por sua vez, reduzir a superfície de ataque do próprio modelo e melhorar métricas de desempenho essenciais, como a precisão e a redução de falsos positivos.

Dessa forma, este TCC busca demonstrar, por meio de uma análise comparativa, que a explicabilidade transcende o papel de um mero requisito de usabilidade. Ela se apresenta como um fator estratégico e indispensável para a otimização da eficácia e da robustez dos

sistemas de segurança cibernética modernos, capacitando especialistas a aprimorar tanto os modelos de IA quanto as estratégias para defesa.

Objetivos

Para alcançar este propósito, foram elaborados os seguintes objetivos:

- **Objetivo Geral:** Investigar como ferramentas de XAI podem ser utilizadas para interpretar, otimizar e avaliar a robustez de modelos de ML na detecção de ataques cibernéticos.
- **Objetivos Específicos:**
 1. Treinar um modelo de classificação para detecção de intrusão utilizando um conjunto de dados público e consolidado.
 2. Aplicar ferramentas de XAI (LIME e SHAP) para analisar as previsões do modelo e extrair a importância relativa de cada característica.
 3. Propor e implementar modificações no modelo com base nas recomendações geradas pela análise de explicabilidade, visando a simplificação e otimização.
 4. Treinar um novo modelo, otimizado, sobre o conjunto de dados modificado.
 5. Realizar uma análise comparativa do desempenho entre o modelo original e o modelo otimizado, com foco em métricas como precisão, F1-score e a taxa de falsos positivos.
 6. Discutir os resultados, avaliando o impacto da explicabilidade na eficácia do modelo e as implicações da transparência para a segurança da própria IA e oferecer um fluxo de trabalho para realização dessa tarefa.

O restante deste documento está organizado da seguinte forma: o Capítulo 1 apre-

senta os conceitos teóricos fundamentais sobre redes, detecção de intrusão, aprendizado de máquina e explicabilidade. O Capítulo 2 detalha um arcabouço para compreensão e otimização de modelos utilizados em IDS. O Capítulo 3 aplica este arcabouço para modelos e *datasets* relacionados à área de cibersegurança e detecção de intrusões, enquanto o Capítulo 4 conclui o trabalho.

Capítulo 1

Conceitos Básicos

Este capítulo estabelece os conceitos fundamentais que servem como alicerce para o desenvolvimento deste trabalho. A discussão inicia-se com uma visão geral dos ataques em redes de computadores, classificando-os e descrevendo suas principais táticas. Em seguida, são detalhados os mecanismos de tratamento de intrusão, com foco nos IDSs e suas metodologias. O capítulo explora como o Aprendizado de Máquina se tornou uma ferramenta essencial para a detecção de anomalias, superando limitações tradicionais. Finalmente, aborda-se o desafio crucial da explicabilidade em modelos complexos de ML, introduzindo o campo da Inteligência Artificial Explicável e as técnicas LIME e SHAP, centrais para a análise realizada nesta monografia.

1.1 Ataques em Redes de Computadores

Na área da tecnologia da informação moderna, a proteção de redes de computadores é um pilar fundamental. A crescente dependência de sistemas interconectados expõe indivíduos e organizações, sejam públicas ou privadas, a uma vasta quantidade de ataques, que podem levar a consequências catastróficas, sejam de cunho reputacional, informacional ou regulatório. Um ataque em redes de computadores pode ser definido como qualquer

tentativa de explorar algum tipo de vulnerabilidade para acessar, alterar, expor, destruir indevidamente qualquer tipo de ativo de um sistema ligado a uma rede.

Os ataques podem ser classificados de várias formas. Uma delas divide-os em passivos e ativos [STALLINGS, 2013](#).

- **Ataques Passivos:** O objetivo é monitorar e coletar o máximo de informações possíveis sobre o sistema alvo sem interferir em sua operação, com o objetivo principal de dificultar a detecção, já que não altera nenhum tipo de dado ou estado da rede. Um dos exemplos inclui a varredura de portas e a interceptação de pacotes. Devido à sua dificuldade de detecção, sua defesa reside na prevenção.
- **Ataques Ativos:** Envolve modificação no fluxo de dados ou a criação de fluxos, o que altera o estado do sistema. São mais fáceis de detectar, apesar de causarem danos, em geral, mais significativos comparados ao anterior. Seu principal mecanismo de defesa é detectar e mitigar esses ataques em tempo.

Eles também podem ser classificados baseados em seu objetivo final [STALLINGS e BROWN, 2014](#). As principais classes incluem, mas não se limitam a:

- **Ataques de Reconhecimento:** fase inicial onde o atacante busca coletar o máximo de informações sobre a rede alvo, como a topologia, sistemas utilizados, serviços em execução ou vulnerabilidades conhecidas.
- **Ataques de Acesso não autorizado:** Tentativa de obter acesso não autorizado a um sistema ou a seus dados. Inclui técnicas como engenharia social, exploração de vulnerabilidades de software (*exploits*) e ataques de força bruta para adivinhar senhas fracas.
- **Ataques de Negação de Serviço:** conhecidos como ataques DoS, o objetivo é tornar um serviço ou recurso da rede indisponível para seus usuários legítimos. Isso é comumente alcançado ao sobrecarregar o sistema alvo com um alto volume de tráfego. Uma variação poderosa é sua variante distribuída, em que o tráfego é

originado de múltiplas fontes comprometidas, o que torna a mitigação mais complexa.

- **Ataque de *Man in the Middle*:** o atacante se posiciona secretamente entre duas partes da rede que estão se comunicando, interceptando e, possivelmente, alterando a comunicação para seu interesse. O atacante pode roubar credenciais, informações sensíveis ou injetar dados maliciosos no fluxo da comunicação.

1.2 Tratamento de Intrusão

Uma intrusão é qualquer conjunto de ações que tentam comprometer algum dos pilares da segurança da informação: confidencialidade, disponibilidade ou integridade de um ativo computacional, seja em rede de computadores ou não. Para combater tais ameaças, foram desenvolvidos os IDSs e os Sistemas de Prevenção de Intrusão (IPS). Um IDS ou o IPS é um dispositivo de hardware ou aplicação de software que monitora o tráfego de uma rede ou as atividades de sistemas de software em busca de atividades maliciosas ou violações de políticas de segurança.

A principal diferença entre os dois sistemas está na atuação em possíveis intrusões. Enquanto o IDS é mais ativo, atuando diretamente no alerta para ação de resposta por parte de um administrador de sistemas ou por um sistema automatizado, o IPS possui ação preventiva, ao recomendar ações para o administrador do sistema, como atualização de versões de sistemas de *software* com vulnerabilidades conhecidas, configurações exploráveis, etc. O foco dessa monografia será ao IDS.

Os IDSs são classificados com base em duas características principais: a fonte de dados e a metodologia de detecção [KHRAISAT et al., 2019](#).

Quanto à fonte de dados, eles podem ser:

- **IDS Baseado em Host (HIDS):** Reside em um único computador (*host*) e monitora as atividades internas desse sistema, como chamadas de sistema, logs de aplicação, modificações em arquivos e tráfego de rede exclusivo do sistema. É eficaz na detecção

de atividades maliciosas que não geram um padrão de tráfego de rede fora do comum.

- **IDS Baseado em Rede (NIDS):** É posicionado em um ponto estratégico da rede para analisar o tráfego que passa por ele, monitorando múltiplos sistemas simultaneamente. Um NIDS inspeciona os pacotes de rede em busca de assinaturas de ataques conhecidos ou tráfego fora do comum da rede.

Quanto à **metodologia de detecção**, as duas abordagens principais são:

- **Detecção por Assinatura:** Esta abordagem utiliza um banco de dados de assinaturas de ataques conhecidos. Cada assinatura representa um padrão associado a uma ameaça específica. O IDS compara o tráfego de rede ou a atividade do sistema com essas assinaturas. Sua principal vantagem é a alta precisão na detecção de ataques conhecidos, com uma baixa taxa de falsos positivos. A grande desvantagem é a sua incapacidade de detectar ataques novos ou desconhecidos, conhecidos como ataques de dia zero.
- **Detecção por Anomalia:** Esta abordagem primeiro estabelece um perfil de comportamento normal da rede ou do sistema, chamado de *baseline*. Qualquer desvio significativo desse perfil é sinalizado como uma possível intrusão. O *baseline* pode ser construído usando técnicas estatísticas ou, mais recentemente, algoritmos de aprendizado de máquina. A principal vantagem é a capacidade de detectar ataques novos e desconhecidos. A desvantagem é a possibilidade de uma taxa mais elevada de falsos positivos, pois um comportamento legítimo, mas atípico, pode ser classificado incorretamente como malicioso.

1.3 Aprendizado de Máquina

Diante dos desafios apresentados pelas abordagens tradicionais de detecção de intrusão, especialmente a incapacidade dos sistemas baseados em assinatura de identificar ataques novos e desconhecidos, o Aprendizado de Máquina emergiu como uma solução poderosa.

A capacidade dos algoritmos de ML de aprender padrões complexos a partir de grandes volumes de dados de rede os torna ideais para a construção de sistemas de detecção por anomalia mais robustos e adaptativos. Como resultado, muitos IDSs modernos já incorporam modelos de aprendizado de máquina para aprimorar sua eficácia, permitindo a identificação de ameaças cada vez mais sofisticadas com maior precisão, como o DarkTrace, um IDS e IPS (são módulos separados) que nasceu inteiramente baseado em técnicas de Inteligência Artificial em escala corporativa *DarkTrace s.d.*

Aprendizado de Máquina é um subcampo da inteligência artificial que se concentra no desenvolvimento de algoritmos que permitem aos computadores aprender a partir de dados e melhorar seu desempenho em uma tarefa específica sem serem explicitamente programados para isso. Em vez de seguir regras predefinidas, um modelo de ML identifica padrões nos dados de treinamento e utiliza esses padrões para fazer previsões ou tomar decisões sobre novos dados.

No contexto da detecção de intrusão, o aprendizado de máquina é promissor para a implementação de sistemas de detecção por anomalia. Modelos de ML podem analisar grandes volumes de dados de tráfego de rede para construir *baselines* complexos e robustos do comportamento normal, superando as limitações de métodos estatísticos tradicionais.

As abordagens de ML são geralmente divididas em quatro categorias principais *ABU-MOSTAFA et al., 2012*:

- **Aprendizado Supervisionado:** Nesta abordagem, o algoritmo é treinado com um conjunto de dados previamente rotulado, onde cada exemplo de entrada possui uma saída ou classe correspondente. O objetivo é que o modelo aprenda uma função de mapeamento que preveja a saída para novos dados de entrada. No contexto de um IDS, os dados de treinamento consistiriam em tráfego de rede rotulado como "normal" ou "ataque" ou rotular o tipo de ataque individualmente.
- **Aprendizado Não Supervisionado:** O algoritmo é treinado com dados não rotulados. O objetivo é encontrar estruturas ou padrões ocultos nos dados. Para a detecção

de intrusão, essa abordagem é útil para identificar anomalias, agrupando o tráfego de rede em *clusters*. A premissa é que o tráfego malicioso formará *clusters* distintos do tráfego normal.

- **Aprendizado por Reforço:** Um agente de software aprende a tomar ações em um ambiente para maximizar uma recompensa cumulativa. O aprendizado ocorre por tentativa e erro. É menos comum em IDSs tradicionais.

Para ilustrar essas abordagens, podemos aplicá-las aos tipos de ataques citados anteriormente:

- **Exemplo de Aprendizado Supervisionado (Acesso não autorizado):** Para detectar um **Ataque de Acesso não autorizado** como Força Bruta, o modelo é treinado com um conjunto de dados onde milhões de conexões são rotuladas como "Normal" ou "Força Bruta". O modelo aprende que a característica "alta frequência de falhas de login da mesma origem" está fortemente associada ao rótulo "Força Bruta", principalmente se for detectado intervalos de tempo não humanos (a cada 5 segundos precisos, por exemplo). Em produção, quando o IDS observa um padrão similar, ele o classifica como ataque (Figura 1.1).

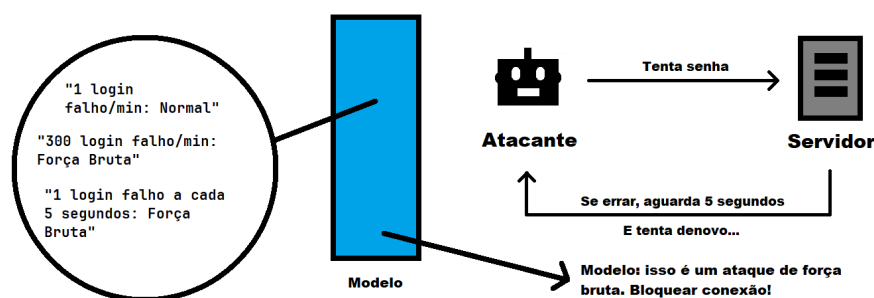


Figura 1.1: Ilustração de um ataque de Força Bruta.

- **Exemplo de Aprendizado Não Supervisionado (Reconhecimento):** Para detectar um **Ataque de Reconhecimento** (como Escaneamento de Portas), o modelo aprende o *baseline* a partir do tráfego normal da rede, que se concentra nas portas 80 e 443. Quando um IP externo tenta se conectar a dezenas de portas diferentes (ex:

21, 22, 23, 25, 3306) em sequência, esse comportamento não se encaixa no "normal" e é sinalizado como uma anomalia (Figura 1.2).

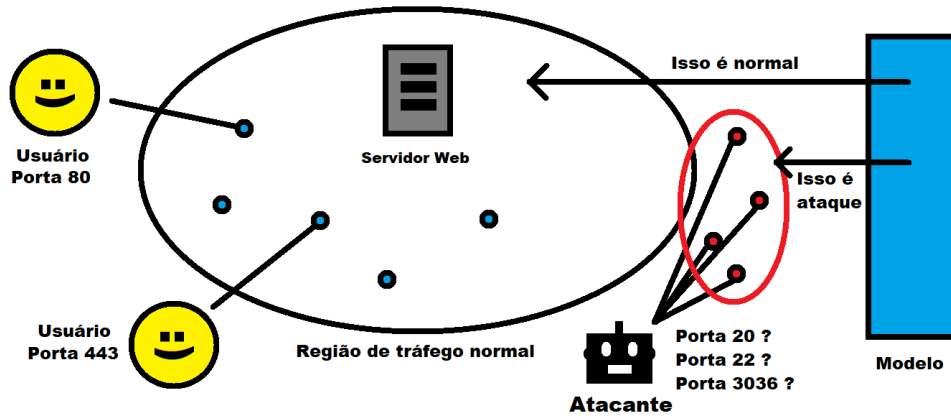


Figura 1.2: Ilustração de um Ataque de Reconhecimento (Escaneamento de Portas).

- **Exemplo de Aprendizado por Reforço (Negação de Serviço):** Esta abordagem é ideal para a **resposta** automática a um **Ataque de Negação de Serviço**. O agente (um IPS ou *firewall*) monitora o estado da rede (ex: carga da CPU do servidor). Quando um ataque DoS começa e a CPU atinge 99% (estado), o agente deve escolher uma ação (ex: bloquear IPs, limitar taxa de conexão). Se ele escolhe "limitar taxa" e a CPU cai para 10%, ele recebe um "sinal positivo", aprendendo que esta é a ação correta para maximizar a disponibilidade do serviço nesse estado (Figura 1.3).

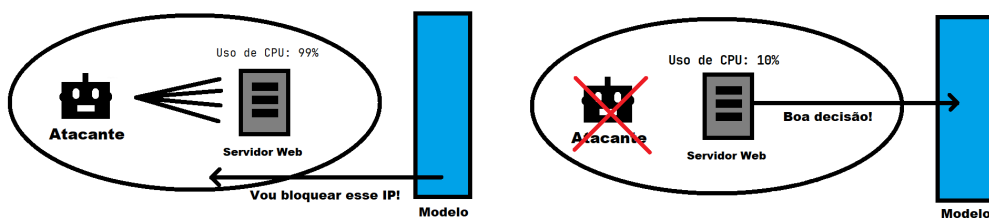


Figura 1.3: Ilustração de um Ataque de Negação de Serviço.

1.4 Explicabilidade

Com a crescente adoção de modelos de aprendizado de máquina em IDS, surge um desafio fundamental: a explicabilidade. Modelos complexos frequentemente operam como

"caixas opacas" [NEUPANE et al., 2022](#). Para um profissional de segurança, um alerta de um IDS com a simples saída classificando o tráfego como malicioso ou não, ou qual o tipo de ataque, é insuficiente. Devido a isso, o analista pode enfrentar dificuldades, como:

- **Validar o Alerta:** É um verdadeiro positivo que exige uma resposta imediata ou um falso positivo que pode ser descartado? A investigação de falsos positivos consome tempo e recursos valiosos.
- **Entender a Ameaça:** Se o alerta for legítimo, quais características do tráfego o tornaram suspeito? Entender o vetor do ataque é o primeiro passo para uma resposta a incidentes eficaz.
- **Confiar no Sistema:** A confiança humana é um fator crucial para a adoção de qualquer ferramenta. Analistas são mais propensos a utilizar e confiar em sistemas cujas decisões eles conseguem compreender e verificar, principalmente em contextos críticos, como hospitais ou bancos.

A **Inteligência Artificial Explicável (Explainable AI - XAI)** é a área que busca resolver esse problema. Seu objetivo é desenvolver técnicas que forneçam o porquê de um modelo ter decidido, transformando as previsões em informações que permitem ao analista de segurança entender a natureza da ameaça, justificar ações de resposta e, em última instância, confiar nas ferramentas de IA que utiliza.

1.4.1 LIME

LIME, ou *Local Interpretable Model-agnostic Explanations* [RIBEIRO et al., 2016](#), é uma técnica de explicabilidade que fornece explicações locais para previsões de qualquer classificador caixa opaca. A intuição por trás do LIME é que, embora um modelo possa ser complexo, é mais fácil aproximá-lo na vizinhança de uma previsão específica usando um modelo mais simples e interpretável, como um modelo linear. Por isso, o LIME pode ser aplicado em uma grande variedade de modelos.

Na prática, para um analista de segurança, ao receber um alerta sobre uma conexão de rede específica, o LIME forneceria uma explicação local e simplificada. A saída seria uma lista legível das características que mais influenciaram a decisão do modelo. Por exemplo, uma conexão foi classificada como maliciosa porque a duração da conexão foi zero ou o número de bytes de origem foi anormalmente alto. Com essa informação concisa, o profissional pode rapidamente contextualizar o evento e julgar se a combinação desses fatores realmente representa uma ameaça ou se corresponde a um comportamento atípico, mas legítimo, da rede.

1.4.2 SHAP

SHAP, ou *SHapley Additive exPlanations* [LUNDBERG e LEE, 2017](#), é uma abordagem de explicabilidade unificada baseada na teoria dos jogos para explicar as previsões de modelos de aprendizado de máquina. O SHAP atribui a cada característica um valor de importância para uma determinada previsão, chamado de **valor SHAP**, que representa a contribuição marginal daquela característica para a decisão do modelo, o que também o permite em ser aplicado a vários tipos de modelos.

Do ponto de vista de um profissional de segurança, o SHAP oferece uma visão geral. Para uma **análise local** de um alerta, ele pode formar um gráfico que evidencia visualmente quais características tenderam a previsão para ataque e quais a levaram para tráfego normal, quantificando a contribuição de cada uma. Isso fornece uma análise de evidências intuitiva para validar (ou invalidar) um alerta.

Contudo, a grande vantagem do SHAP reside em sua capacidade de fornecer explicações globais. O analista pode agregar os valores SHAP de milhares de previsões para entender quais características o modelo de IDS considera mais importantes, em geral. Isso é extremamente útil para:

- **Auditoria do Modelo:** O modelo está focando em características que fazem sentido do ponto de vista da segurança (ex: flags TCP, portas incomuns) ou está se baseando

em correlações ilegítimas entre os dados?

- **Inteligência de Ameaças (*Threat Intelligence*):** A análise global pode revelar que uma característica anteriormente sem importância está agora entre as principais indicadoras de ataque, sugerindo uma nova tática ou vetor de ameaça sendo explorado por atacantes.

Capítulo 2

Arcabouço para Compreensão e Otimização de Modelos de Detecção de Intrusão

A aplicação de IA em sistemas de cibersegurança tornou-se uma prática comum, com modelos de Detecção de Intrusão alcançando níveis de acurácia sem precedentes. No entanto, o avanço destes modelos, especialmente os baseados em Aprendizagem Profunda (*Deep Learning*), introduziu um desafio significativo: a opacidade. Tais sistemas são frequentemente tratados como "caixas-opacas", pois não fornecem uma justificativa para suas previsões [NEUPANE et al., 2022](#).

Para um especialista em cibersegurança, um alerta de um IDS é somente o ponto de partida de uma investigação. A falta de contexto — o "porquê" por trás de uma detecção — cria uma barreira para os analistas do Centro de Operações de Segurança (SOC), impedindo uma tomada de decisão ágil e a construção de confiança no sistema. A questão fundamental transcende a simples classificação: a detecção foi baseada em um padrão de ataque conhecido ou em uma anomalia sutil que pode indicar uma nova tática adversária?

Este capítulo apresenta um arcabouço metodológico para "interrogar" um modelo de IDS, transformando suas decisões de caixa opaca em decisões claras e verbosas, ou seja, entendíveis por um especialista de cibersegurança. O objetivo não é somente validar o modelo, mas usá-lo como uma ferramenta de análise para aprimorar a inteligência de ameaças (*threat intelligence*) e otimizar a eficiência operacional do próprio sensor de detecção, abordando diretamente a necessidade de projetar Sistemas de Detecção de Intrusão Explicáveis.

2.1 O Arcabouço

O fluxo a seguir foi desenhado para ser executado por um analista de segurança, para extrair o máximo de valor prático das técnicas de XAI.

2.1.1 Estabelecendo o Modelo de Referência

O ponto de partida é treinar um modelo de aprendizado de máquina com um conjunto de dados de referência, utilizando-se de todas as *features* disponíveis. A literatura demonstra um trade-off entre a acurácia e a interpretabilidade dos modelos NEUPANE *et al.*, 2022. Portanto, a escolha de um modelo complexo, como uma rede neural¹ ou um ensemble,² é justificada pela necessidade de alta precisão na detecção, um requisito fundamental em IDS NEUPANE *et al.*, 2022. Este modelo inicial servirá como *baseline* de desempenho e complexidade.

Neste ponto, o modelo é uma caixa opaca. Ele pode ter uma acurácia de 99%, mas sua lógica interna é inacessível. Ele pode estar detectando ataques pelos motivos corretos ou pode ter aprendido "atalhos" ou correlações espúrias, como associar o IP de origem a um ataque, em vez do comportamento do tráfego. A confiança neste modelo ainda é baixa. A

¹ Uma rede neural é um modelo de aprendizado de máquina inspirado no cérebro humano, composto por camadas de nós (neurônios) interconectados. É altamente eficaz em aprender padrões complexos, sendo um exemplo comum de modelo caixa opaca.

² Um *ensemble* (ou "conjunto") é uma técnica que combina as previsões de múltiplos modelos de aprendizado de máquina para criar um modelo final mais robusto e preciso.

missão nas próximas etapas é construir essa confiança e validar sua lógica.

2.1.2 Entendendo as Decisões do Modelo

Nesta fase, utilizamos LIME e SHAP, as duas técnicas de explicação pós-hoc. Elas investigam a lógica do modelo a partir de duas perspectivas complementares: a análise forense de um único evento (local) e o mapeamento do comportamento geral do adversário (global).

O LIME responde à pergunta: *"Por que este alerta específico foi gerado?"*. A capacidade de explicar uma única predição é definida como explicabilidade local e é fundamental para a análise de causa raiz de um evento de segurança [NEUPANE et al., 2022](#). Os passos dessa fase relacionados ao LIME são os seguintes:

1. Selecione uma instância específica do seu conjunto de teste. Caso não haja o interesse em um resultado específico, é possível aplicá-lo para todos os tipos de classificações possíveis.
2. Aplique o LIME para gerar uma explicação para essa predição.
3. Analise as *features* que mais contribuíram para a decisão. O LIME mostrará as principais evidências que levaram o modelo a tomar aquela decisão.

A Figura 2.1 ilustra a saída visual do terceiro passo anterior, embora aplicada a um problema de diagnóstico médico. O modelo previu "Has diabetes"(tem diabetes) com 73% de confiança. O gráfico central, que é a explicação do LIME, identifica as "provas" para essa decisão: a barra azul ('Glucose > 99.00') é a principal evidência *a favor* da previsão, enquanto as barras laranjas ('Age < 40.00', 'Pregnancies > 6.00') são as evidências *contra*. No contexto de um IDS, um analista veria, da mesma forma, quais *features* da rede (ex: 'porta de destino' ou 'duração da conexão') foram as principais "provas" que levaram a um alerta de ataque.

O LIME é interessante para validar detecções críticas. Se o modelo classificou um fluxo como um vazamento de dados, o LIME deveria destacar *features* relacionadas ao tamanho

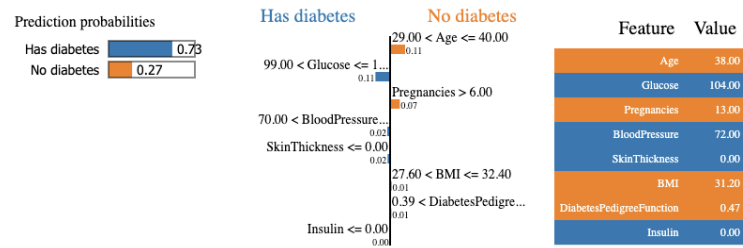


Figura 2.1: Exemplo de saída gerada pelo LIME, retirado de <https://towardsdatascience.com>

do pacote. Se, em vez disso, ele destacar *features* irrelevantes, o modelo pode estar usando uma heurística fraca, e o alerta pode ser um falso positivo.

Já o SHAP responde à pergunta: "O que, em geral, o modelo considera ser um ataque?". Ao agregar as explicações de múltiplas instâncias, o SHAP fornece uma visão global, revelando a forma com que o modelo aprendeu para cada classe de ataque. Os passos dessa fase relacionados ao SHAP são os seguintes:

1. Aplique o SHAP a um subconjunto representativo dos seus dados de teste.
2. Gere um gráfico de resumo (*summary plot*) que mostra as *features* mais importantes globalmente.
3. Analise os gráficos de dependência para entender como o valor de uma *feature* impacta a probabilidade de um ataque.

A Figura 2.2 ilustra uma das possíveis saídas visuais oferecidas pelo SHAP. Neste caso, quanto mais no topo do eixo Y, maior o peso daquela *feature* para a decisão final do modelo, em média. Além disso, os pontilhados no eixo X significam os momentos em que, ao longo do processo de decisão, aquela *feature* tendeu a decisão para o lado positivo ou para o lado negativo. A cor dos pontilhados, tanto no eixo Y quanto no eixo X, indicam se o valor daquela *feature* era alto (vermelho) ou baixo (azul) para uma previsão específica, já que o SHAP é a média de várias destas previsões.

O ranking global do SHAP é a fonte de *threat intelligence*. As *features* no topo são os Indicadores de Comprometimento (IoC) mais fortes que o seu modelo aprendeu. *Features* inesperadas no topo podem indicar uma nova variante de ataque que o modelo capturou.



Figura 2.2: Exemplo de gráfico "espinha de peixe" gerado pelo SHAP, retirado de SHAP (SHapley Additive exPlanations) *s.d.*

2.1.3 Extração e Validação da Lógica

As explicações geradas devem auxiliar os analistas em tarefas como mitigação de ataques, coleta de inteligência e análise forense. Para isso, o analista deve seguir a análise de três formas:

1. **Validação de Conhecimento:** Verificar se a lógica do modelo corresponde ao esperado. Isso ajuda a confirmar que o modelo está funcionando corretamente.
2. **Descoberta de Novas Táticas:** É necessário verificar se o modelo está utilizando *features* que não consideradas importante. Uma *feature* como a média de tempo entre requisições sendo relevante para um ataque de força bruta pode indicar que o atacante está usando pausas estratégicas para evitar a detecção.
3. **Análise de Pontos Cegos e Evasão:** Verificar quais são as *features* mais importantes para o tráfego classificado como benigno e se um atacante poderia manipular essas *features* para mascarar um ataque. As próprias explicações podem se tornar um novo vetor de ataque para atores maliciosos NEUPANE *et al.*, 2022.

2.1.4 Otimização da Detecção

Ao saber o peso de cada *feature* em uma decisão, é possível otimizar o modelo. Este passo é crucial, pois o desempenho é de extrema importância para um IDS, e as explicações não devem atrasar o sistema desnecessariamente NEUPANE et al., 2022. Métodos como o SHAP podem ser computacionalmente caros e não funcionar em tempo real, tornando inviável sua aplicação direta em produção para cada predição NEUPANE et al., 2022. O processo envolve treinar novas versões do modelo, cada uma com um subconjunto reduzido de *features*, com base no ranking global do SHAP, capturando as métricas necessárias para comparar com a *baseline*.

Modelo	F1-Score	Tamanho MB
Baseline (80 <i>features</i>)	0.995	120.4
Otimizado (45 <i>features</i>)	0.994	75.8
Agressivo (25 <i>features</i>)	0.981	42.1

Tabela 2.1: Exemplo de análise de trade-off para o modelo otimizado. O F1-Score representa o equilíbrio entre detectar ameaças (*recall*) e evitar falsos alarmes (*precisão*).

A etapa final é medir o impacto da otimização, focando no trade-off entre o desempenho de detecção e os ganhos de eficiência, principalmente ao considerar ambientes locais (*on-premises*). Isso se traduz em economia de recursos (CPU, memória, disco) e um SOC (*Security Operations Center*) mais ágil. No caso da Tabela 2.1, é mostrado um exemplo hipotético de um modelo que possuía 80 *features* originalmente. Após a execução, o Analista deve ter em mente, baseando-se nos resultados encontrados e com o objetivo final para aquele modelo, qual será o *trade-off* que ele está disposto a realizar ao priorizar as métricas para detecção ou priorizar métricas de desempenho. Por exemplo, caso ele tenha um ambiente computacionalmente limitado, faz sentido diminuir a dimensão do modelo para ter um desempenho satisfatório, mesmo que isso signifique ter um F1-Score menor. No caso da Tabela 2.1, pode-se considerar que a adoção do modelo Otimizado em detrimento do Baseline faz sentido, pois a diminuição da métrica de detecção a partir do F1-Score

compensa a diminuição drástica de tamanho.

2.2 Recapitulação

A integração de XAI no ciclo de vida de um IDS transcende a mera validação de modelos. Ela capacita o analista de segurança a:

- **Confiar e justificar** as decisões da IA perante outras equipes, construindo o que a literatura chama de "confiança e dependência apropriadas" no sistema [NEUPANE et al., 2022](#).
- **Descobrir** padrões de ataque sutis e emergentes.
- **Otimizar** a infraestrutura de segurança, tornando-a mais enxuta e responsiva.

Em suma, o arcabouço aqui proposto utiliza XAI para transformar o modelo de IDS de uma caixa opaca de detecção em um parceiro transparente na caça e análise de ameaças.

Capítulo 3

Análise de Desempenho

Nesta seção, são apresentados os modelos e conjuntos de dados selecionados para a análise experimental. O processo de seleção é descrito, seguido por uma descrição detalhada dos *datasets* e do modelo utilizado neste estudo, o OCPP-IDS.

3.1 Modelos Utilizados

A primeira etapa prática deste trabalho consistiu na identificação de modelos de detecção de intrusão e seus respectivos conjuntos de dados (*datasets*) que fossem, ao mesmo tempo, relevantes para o estado da arte em segurança cibernética e acessíveis para replicação e análise. Para isso, foi realizada uma busca no repositório IEEE DataPort,¹ uma plataforma de dados de pesquisa científica.

A pesquisa inicial, conduzida em 26 de março de 2025, utilizou o filtro de categoria *Security*, resultando em 34 páginas de resultados, totalizando centenas de conjuntos de dados. Cada resultado foi metodicamente analisado para verificar sua adequação ao escopo da monografia, resultando em uma lista primária de 10 candidatos que abordavam a detecção de ataques em redes.

¹ <https://ieee-dataport.org/datasets>

Em uma segunda fase de análise, investigou-se a disponibilidade de um modelo de aprendizado de máquina associado a cada um desses 10 *datasets*. Para os casos em que nem o modelo treinado nem um *script* para sua criação estavam publicamente disponíveis, os autores dos artigos foram contatados por e-mail. Nenhuma solicitação enviada por e-mail foi atendida. A principal razão fornecida pelos demais pesquisadores para a não divulgação foi a confidencialidade dos artefatos, um desafio recorrente na área de segurança que pode dificultar a reprodutibilidade de experimentos. Após essa etapa, restaram apenas dois *datasets* nos quais cada um deles tinham um modelo atrelado.

3.1.1 OCPP-IDS

O *Open Charge Point Protocol* (OCPP) é um protocolo de comunicação da camada de aplicação, de especificação aberta, que se tornou o padrão *de facto* para a gestão de infraestrutura de carregamento de veículos elétricos. Sua principal função é garantir a interoperabilidade entre as Estações de Carregamento de Veículos Elétricos (EVCS), popularmente conhecidas como *chargers*, e os Sistemas Centrais de Gestão (CSMS), que são os *softwares* de controle. Operando sobre conexões, o OCPP padroniza operações essenciais como autorização de usuários, início e término de sessões de carregamento, monitoramento remoto e atualizações.

Contudo, por ter sido projetado inicialmente com foco na interoperabilidade e não na segurança, o protocolo, especialmente em sua versão 1.6, possui vulnerabilidades inerentes. A comunicação entre a estação e o sistema central pode não ser adequadamente criptografada ou autenticada, tornando a infraestrutura um alvo para diversos tipos de ciberataques, como *Man-in-the-Middle* (MITM), Negação de Serviço e manipulação de dados. O *dataset* DALAMAGKAS *et al.*, 2025 foi criado precisamente para estudar essas ameaças, a partir da emulação de um ambiente realista onde múltiplos ataques foram executados contra uma infraestrutura baseada no OCPP 1.6.

Para gerar um *dataset* realista, contendo tanto tráfego de rede benigno quanto malicioso,

foi montado um ambiente de laboratório controlado pelos autores do *dataset*. A topologia da rede, mostrada na Figura 3.1, foi projetada para simular uma interação real entre os componentes de um ecossistema de carregamento de VE e um ator malicioso, sendo composta por quatro elementos principais:

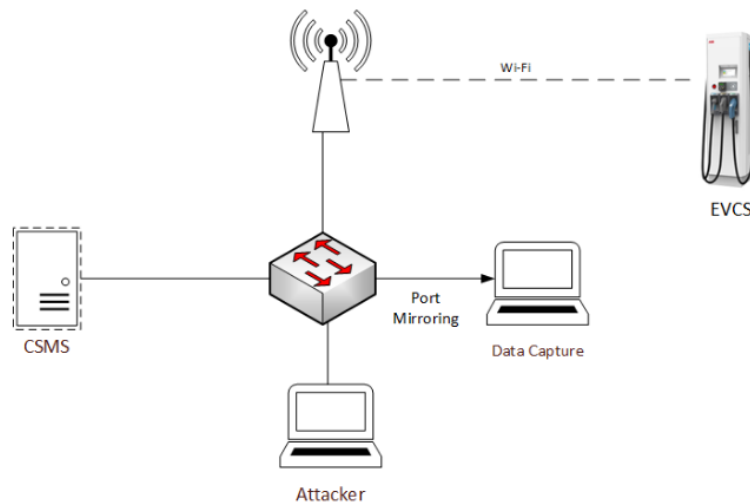


Figura 3.1: Topologia de rede utilizada para criar o *dataset*, retirado de *DALAMAGKAS et al., 2025*

- **Estação de Carregamento de VE (‘EVCS’):** Um carregador físico, que representa o equipamento de ponta no sistema e uma das vítimas dos ataques.
- **Sistema de Gestão (‘CSMS’):** Um servidor responsável por controlar e gerenciar a estação de carregamento, sendo o alvo principal dos ataques de negação de serviço.
- **Atacante (‘Attacker’):** Uma máquina equipada com ferramentas de cibersegurança, posicionada na rede para interceptar a comunicação e lançar os ataques contra o EVCS e o CSMS.
- **Capturador de Dados (‘Data Capture’):** Um computador configurado em modo promíscuo (um modo que permite à placa de rede capturar todos os pacotes que passam por ela, mesmo que não sejam destinados a ela), com a técnica de *port mirroring*,² permitindo-lhe monitorar e gravar uma cópia de todo o tráfego que

² O *Port mirroring* (espelhamento de porta) é uma funcionalidade de *switches* de rede que copia todo o tráfego de pacotes de uma ou mais portas para uma porta de monitoramento específica, onde o dispositivo de captura está conectado.

circula na rede.

Neste ambiente, o atacante executou uma série de ataques predefinidos, enquanto o sistema de captura registrava todas as comunicações em arquivos de formato .pcap, uma extensão para armazenar tráfego capturado de uma rede. O resultado dessa etapa foi um *dataset* bruto contendo o registro completo das interações de rede durante operações normais e sob ataque. Dentro deste ambiente experimental, foram executados quatro tipos de ataques distintos, cada um explorando diferentes vulnerabilidades do protocolo OCPP e da infraestrutura de rede:

- **Manipulação do Perfil de Carregamento (*Charging Profile Manipulation attack*):** Este ataque visa instalar perfis de carregamento maliciosos nas estações para forçar um consumo de energia superior ao permitido. Utilizando uma técnica de MITM com envenenamento de ARP (*Address Resolution Protocol*), o que significa que o atacante envia mensagens falsas na rede para enganar os dispositivos e fazê-los acreditar que ele é o destinatário legítimo da comunicação, o atacante intercepta o pacote que define a forma de carregamento e altera o atributo `limit` para um valor mais alto.
- **Negação de Carregamento via Manipulação de `idTag` (*Denial of Charge via id-Tag Manipulation attack*):** O objetivo é impedir que o CSMS autorize novas sessões de carregamento. Através de um ataque MITM, o atacante modifica o identificador de usuário (`idTag`) nos pacotes por um valor aleatório e inválido. Consequentemente, o CSMS rejeita a transação, causando uma negação de serviço ao usuário.
- **Inundação de *Heartbeat* (*Heartbeat Flooding DoS*):** Este ataque de força bruta visa exaurir os recursos do CSMS. O atacante simula 100 estações de carregamento que, após se conectarem ao servidor, começam a enviar mensagens de *Heartbeat* (um tipo de mensagem utilizada para verificar se o outro lado da conexão está ativo) em uma frequência altíssima (a cada 250 ms). O volume massivo de mensagens sobrecarrega o servidor, levando a uma condição de Negação de Serviço.

- **Acesso Não Autorizado (*Unauthorized Access*):** Com um duplo objetivo, este ataque busca obter acesso não autorizado ao CSMS via força bruta e, ao mesmo tempo, exaurir seus recursos. O atacante implanta 100 clientes que tentam se conectar simultaneamente ao servidor utilizando identificadores de estação (*charge point IDs*) gerados aleatoriamente. O grande volume de tentativas de autenticação sobrecarrega o sistema, resultando em DoS e aumentando a probabilidade de uma credencial ser descoberta.

Com os dados brutos de tráfego capturados em formato *.pcap*, a etapa seguinte realizada pelos autores do *dataset* consistiu em processá-los para extrair características (*features*) relevantes que pudessem ser utilizadas por modelos de aprendizado de máquina. Para transformar o tráfego não estruturado em um formato para análise (*.csv*), foram empregadas duas ferramentas complementares que analisam o tráfego sob diferentes perspectivas:

- **CICFlowMeter:** Uma ferramenta que analisa o tráfego na camada de transporte (TCP). Ela agrupa os pacotes em fluxos de comunicação e calcula mais de 80 características estatísticas para cada fluxo, como duração, volume de bytes, tamanho dos pacotes, etc. O CICFlowMeter oferece uma visão comportamental do tráfego, sem analisar o conteúdo específico dos protocolos de aplicação.
- **OCPFlowMeter:** Uma ferramenta especializada, desenvolvida pelos próprios autores do *dataset*, que analisa o tráfego na camada de aplicação. Diferentemente do CICFlowMeter, ela "entende" o protocolo OCPP 1.6, sendo capaz de dissecar as mensagens para extrair características específicas, como os tipos de comandos OCPP, a frequência de determinadas operações e outros atributos semânticos.

A combinação estratégica dessas duas ferramentas permitiu a criação de dois *datasets* a partir de um tempo limite de fluxo de 120 segundos. Cada linha contém tanto as características estatísticas da camada de transporte, geradas pelo CICFlowMeter, com 85 *features*, e características contextuais da camada de aplicação, extraídas pelo OCPFlowMeter, com 49

features. Essa visão integrada é fundamental para treinar modelos de IA capazes de aprender as nuances que diferenciam o tráfego legítimo do malicioso no contexto específico do carregamento de veículos elétricos.

O uso de Aprendizado Federado é um pilar central deste projeto, uma vez que o próprio *dataset* foi criado para dar suporte a essa técnica. Essa é uma abordagem de treinamento de modelos que ganha destaque, especialmente em cenários de Internet das Coisas [FERRAG *et al.*, 2021](#). Diferentemente das abordagens centralizadas tradicionais, onde todos os dados são coletados em um único servidor para treinamento, o Aprendizado Federado permite que um modelo global seja treinado colaborativamente em múltiplos dispositivos descentralizados (como sensores ou gateways IoT) sem que os dados brutos precisem sair desses dispositivos.

Ao fornecer versões balanceadas dos dados que já vêm divididas em pastas para três "clientes" distintos, o IDS utiliza da estrutura do aprendizado federado, o que faz sentido por ser uma consequência direta da própria infraestrutura de carregamento de veículos elétricos, distribuída, em que o CSMS atua como o servidor central que agrega os aprendizados para construir um modelo de detecção global mais robusto. Além de ser vantajosa para a topologia, também é importante por resolver questões críticas de segurança e privacidade, já que os dados de tráfego podem conter informações sensíveis sobre seus usuários, algo que não precisa sair da estação de carregamento para ser centralizado em um servidor, mitigando parte dos riscos de privacidade [Lu *et al.*, 2020](#).

Embora o treinamento do modelo de detecção tenha sido realizado em FL com múltiplos clientes, a análise de explicabilidade com LIME e SHAP realizada nesta monografia foi intencionalmente focada no modelo treinado de um único cliente. Esta decisão foi tomada devido à natureza das ferramentas de XAI e à complexidade do ambiente federado. LIME e SHAP são técnicas aplicadas a modelos já treinados para explicar suas previsões em relação a um conjunto de dados. Em um cenário federado, isso resultaria em múltiplas explicações distintas — uma para o modelo de cada cliente. Como o conjunto de dados

OCP-IDS foi distribuído de forma balanceada entre os clientes, é esperado que os modelos de cada cliente apresentassem comportamentos muito semelhantes, o que torna a geração de explicações para cada um deles um processo redundante. Portanto, a análise sobre um único cliente é suficiente para evitar a complexidade e a redundância de comparar múltiplos resultados de explicabilidade que seriam, na maioria, equivalentes.

Quanto à técnica de classificação, o modelo baseia-se em uma Rede Neural Profunda (DNN). De forma simplificada, este modelo atua processando os dados de entrada por múltiplas camadas internas, permitindo que ele aprenda padrões complexos de tráfego que seriam difíceis de identificar manualmente. É importante ressaltar que a arquitetura específica da rede (quantidade de camadas ou neurônios) não é o objeto de estudo deste trabalho, e sim como tornar suas decisões interpretáveis.

Dessa forma, além da restrição ao uso de um cliente único para o treinamento — o que descaracteriza o uso do Aprendizado Federado nesta etapa específica —, modificações foram feitas na estrutura dessa rede neural, aproveitando-se de recursos já implementados anteriormente pelos autores. Tais alterações visaram adaptar o modelo para o arcabouço descrito no Capítulo 2. Ou seja, a partir de agora, independentemente de atualizações no *dataset* ou no código que origina o modelo, o arcabouço poderá ser aplicado quantas vezes o analista julgar necessário, garantindo a reprodutibilidade da análise de explicabilidade.

Para viabilizar a aplicação do SHAP e do LIME, a saída do modelo, que originalmente classificava 4 tipos de ataques possíveis, foi generalizada para uma classificação binária: "Não ataque" e "Ataque". Essa alteração promove uma análise de comportamento mais assertiva, focada na distinção crítica entre tráfego legítimo e malicioso. Como as modificações feitas mostraram-se benéficas para a estrutura do código, após a aprovação desta monografia, elas serão submetidas aos autores originais para contribuição no projeto principal.

3.2 Resultados

Todo o fluxo de trabalho utilizado para configurar os modelos de aprendizado para terem suporte tanto ao LIME quanto ao SHAP está disponível no repositório do GitHub, em <https://github.com/th-duvanel/ids-explainability>.

O repositório foi organizado para armazenar todos os *datasets* estudados e seus respectivos modelos para treinamento, os quais tiveram suas saídas armazenadas no próprio repositório. Entre as saídas capturadas relacionadas à explicabilidade, a partir do *dataset* original:

- Lista das *features* ordenadas por importância de acordo com o LIME
- Lista das *features* ordenadas por importância de acordo com o SHAP
- Arquivos necessários para geração de gráficos SHAP

As métricas capturadas, de cada *dataset* utilizado para o treinamento por 30 rodadas, foram:

- **Tempo para treinamento:** Mede o tempo total, que o algoritmo levou para aprender com os dados de treinamento e construir a versão final do modelo.
- **Acurácia:** Proporção de previsões corretas em relação ao número total de amostras.
- **Precisão:** Informa a porcentagem que é, de fato, um ataque, sempre que o modelo classifica algo como "ataque". Foca em minimizar falsos positivos.
- **Recall:** Mede a proporção de ataques reais que foram corretamente identificados pelo modelo. De todos os eventos que eram, de fato, um ataque, quantos deles o IDS conseguiu classificar corretamente. O objetivo principal desta métrica é minimizar os falsos negativos.
- **F1-Score:** É a média harmônica entre Precisão e Recall.
- **Tamanho do modelo:** Tamanho do modelo em MB (*mega bytes*).

Para proporcionar a reprodutibilidade dos experimentos, no README presente no repositório, existem instruções claras para executar e treinar todos os modelos utilizados e gerar suas métricas. Além disso, também há um arquivo do tipo `.ipynb` para gerar todos os gráficos apresentados nesta monografia.

3.2.1 OCPP-IDS

Peso das *Features*

Ao analisar as tabelas de importância de *features* geradas para os datasets CICFlowMeter (Tabela 3.1) e OCPPFlowMeter (Tabela 3.2), observa-se uma notável divergência entre os rankings atribuídos pelas ferramentas LIME e SHAP. Um exemplo marcante no *dataset* CICFlowMeter é a característica `Bwd_RST_Flags` (a quantidade de pacotes com a flag `RESET`, que é uma forma abrupta de encerrar uma conexão indisponível, devido a, por exemplo, uma porta fechada), classificada em 1º lugar pelo SHAP e em 82º pelo LIME. Similarmente, no *dataset* OCPPFlowMeter, a característica `flow_max_ocpp16_metervalues_wh_diff` (a diferença da medição de energia entre duas mensagens consecutivas de um mesmo fluxo) é a 5ª mais importante para o SHAP, mas apenas a 37ª para o LIME. Essas diferenças são uma consequência direta das distintas abordagens teóricas que cada método emprega para aproximar e explicar o comportamento de um modelo caixa opaca.

A importância das *features* na Tabela 3.1 se alinha com a natureza dos ataques executados no *dataset* OCPP-IDS. Os ataques de **Inundação de Heartbeat** e **Acesso Não Autorizado** são, em essência, ataques de volume e força bruta, projetados para sobrecarregar o CSMS (servidor). O CICFlowMeter captura exatamente o comportamento desse estresse na camada de transporte. Faz todo o sentido que características que medem volume e frequência de pacotes estejam no topo, como `Subflow_Fwd_Packets` (SHAP 2º) e `Total_Fwd_Packet` (LIME 5º), que contam o número de pacotes enviados pelo cliente. Da mesma forma, `Fwd_IAT_Mean` (LIME 8º) e `Fwd_IAT_Min` (LIME 6º), que medem o tempo entre os pacotes do atacante, são cruciais; em um ataque de inundação, esse tempo é

Feature (CICFlowMeter)	LIME	SHAP	Feature (CICFlowMeter)	LIME	SHAP
ACK Flag Count	39	20	Fwd IAT Min	6	35
Active Max	37	50	Fwd IAT Std	17	45
Active Mean	71	37	Fwd IAT Total	72	83
Active Min	15	15	Fwd PSH Flags	53	67
Active Std	54	59	Fwd Packet Length Max	23	71
Average Packet Size	21	72	Fwd Packet Length Mean	76	74
Bwd Act Data Pkts	42	62	Fwd Packet Length Min	83	85
Bwd Bulk Rate Avg	73	4	Fwd Packet Length Std	44	80
Bwd Bytes/Bulk Avg	59	5	Fwd Packet/Bulk Avg	32	13
Bwd Header Length	20	12	Fwd Packets/s	13	22
Bwd IAT Max	57	61	Fwd RST Flags	81	8
Bwd IAT Mean	34	24	Fwd Seg Size Min	27	60
Bwd IAT Min	58	3	Fwd Segment Size Avg	78	75
Bwd IAT Std	64	65	Fwd TCP Retrans. Count	68	73
Bwd IAT Total	75	82	Fwd URG Flags	4	6
Bwd Init Win Bytes	62	69	ICMP Code	84	31
Bwd PSH Flags	11	54	ICMP Type	85	30
Bwd Packet Length Max	35	34	Idle Max	14	33
Bwd Packet Length Mean	31	36	Idle Mean	46	49
Bwd Packet Length Min	79	25	Idle Min	18	32
Bwd Packet Length Std	49	47	Idle Std	48	43
Bwd Packet/Bulk Avg	16	58	PSH Flag Count	12	56
Bwd Packets/s	50	66	Packet Length Max	69	78
Bwd RST Flags	82	1	Packet Length Mean	43	52
Bwd Seg Size Min	80	81	Packet Length Min	45	14
Bwd Segment Size Avg	29	19	Packet Length Std	65	79
Bwd TCP Retrans. Count	51	28	Packet Length Variance	30	16
Bwd URG Flags	3	7	RST Flag Count	63	18
Down/Up Ratio	74	84	SYN Flag Count	66	64
ECE Flag Count	2	23	Subflow Bwd Bytes	25	26
FIN Flag Count	70	76	Subflow Bwd Packets	67	27
FWD Init Win Bytes	24	68	Subflow Fwd Bytes	40	11
Flow Bytes/s	52	38	Subflow Fwd Packets	38	2
Flow Duration	28	53	Total Bwd packets	47	63
Flow IAT Max	55	48	Total Connection Flow Time	41	70
Flow IAT Mean	26	40	Total Fwd Packet	5	55
Flow IAT Min	10	42	Total Length of Bwd Packet	9	29
Flow IAT Std	33	41	Total Length of Fwd Packet	56	77
Flow Packets/s	22	39	Total TCP Retrans. Count	60	51
Fwd Act Data Pkts	36	57	URG Flag Count	1	21
Fwd Bulk Rate Avg	77	9			
Fwd Bytes/Bulk Avg	61	17			
Fwd Header Length	7	10			
Fwd IAT Max	19	46			
Fwd IAT Mean	8	44			

Tabela 3.1: Comparativo de importância de features para o CICFlowMeter (LIME vs. SHAP). As colunas LIME e SHAP representam a posição no ranking de importância nos respectivos métodos de explicabilidade.

intencionalmente reduzido e o intervalo preciso e não humano para sobrecarregar o alvo.

A divergência mais notável, a Bwd RST Flags (SHAP 1º), é talvez a explicação mais lógica de todas do ponto de vista global. Esta *feature* conta os pacotes RST (Reset) enviados *de volta* pelo servidor. Diante de uma inundação de conexões ilegítimas ou tentativas de autenticação falhas (força bruta), a resposta esperada do CSMS é precisamente encerrar abruptamente essas conexões, enviando um volume massivo de pacotes RST. O SHAP, com sua visão global, identifica corretamente esta "reação" do servidor como o indicador de ataque mais confiável. Em contrapartida, a alta classificação de LIME para *flags* como URG

Flag Count (LIME 1º), que representa pacotes urgentes do protocolo TCP e ECE Flag Count (LIME 2º), que é uma forma de avisar sobre congestionamento na rede do protocolo TCP, apesar de serem consideradas mais raras, faz sentido em decisões específicas como as estudadas pelo LIME, principalmente em um contexto de Negação de Serviço.

Feature (OCPPFlowMeter)	LIME	SHAP	Feature (OCPPFlowMeter)	LIME	SHAP
bw_websocket_bytes_per_second	29	1	flow_total_http_2xx_packets	49	21
bw_websocket_packets_per_second	14	2	flow_total_http_4xx_packets	38	48
flow_avg_ocpp16_metervalues_wh_diff	46	16	flow_total_http_5xx_packets	4	24
flow_avg_ocpp16_setchargingprofile_limit	40	45	flow_total_http_get_packets	27	22
flow_avg_ocpp16_setchargingprofile_minchargingrate	35	39	flow_total_ocpp16_authorize_not_accepted_packets	44	47
flow_down_up_ratio	11	28	flow_total_ocpp16_heartbeat_packets	20	32
flow_duration	17	20	flow_total_ocpp16_metervalues	28	13
flow_end_timestamp	26	40	flow_total_ocpp16_remotestarttransaction_packets	42	37
flow_max_ocpp16_metervalues_soc	8	15	flow_total_ocpp16_resetHard_packets	5	8
flow_max_ocpp16_metervalues_wh_diff	37	5	flow_total_ocpp16_resetSoft_packets	47	9
flow_max_ocpp16_setchargingprofile_limit	41	49	flow_total_ocpp16_setchargingprofile_packets	19	29
flow_max_ocpp16_setchargingprofile_minchargingrate	13	12	flow_total_ocpp16_starttransaction_packets	45	46
flow_min_ocpp16_metervalues_soc	7	14	flow_total_ocpp16_unlockconnector_packets	6	10
flow_min_ocpp16_metervalues_wh_diff	48	19	flow_total_websocket_close_packets	36	7
flow_min_ocpp16_setchargingprofile_limit	39	41	flow_total_websocket_data_messages	30	35
flow_min_ocpp16_setchargingprofile_minchargingrate	21	11	flow_total_websocket_ping_packets	32	44
flow_start_timestamp	24	42	flow_total_websocket_pong_packets	9	6
flow_total_ACK_flag	16	33	flow_websocket_bytes_per_second	23	3
flow_total_CWR_flag	2	26	flow_websocket_packets_per_second	22	18
flow_total_ECE_flag	3	25	fw_websocket_bytes_per_second	12	4
flow_total_FIN_flag	18	38	fw_websocket_packets_per_second	31	17
flow_total_PSH_flag	25	30	total_bw_packets	10	31
flow_total_RST_flag	43	23	total_flow_packets	15	34
flow_total_SYN_flag	34	43	total_fw_packets	33	36
flow_total_URG_flag	1	27			

Tabela 3.2: Comparativo de importância de features para o OCPPFlowMeter (LIME vs. SHAP). As colunas LIME e SHAP representam a posição no ranking de importância nos respectivos métodos de explicabilidade.

A análise do OCPPFlowMeter (Tabela 3.2) revela como o modelo aprende a detectar ataques na camada de aplicação. A visão global do SHAP das *features* mais importantes está ligada ao volume de tráfego WebSocket: `bw_websocket_bytes_per_second` (SHAP 1º) (soma de *payloads* por segundo do servidor para o cliente), `bw_websocket_packets_per_second` (SHAP 2º) (número de pacotes WebSocket por segundo do servidor para o cliente) e suas contrapartes no fluxo total e na direção do cliente para o servidor, que são `flow_websocket_bytes_per_second` (SHAP 3º) e `fw_websocket_bytes_per_second` (SHAP 4º). Esta é uma conexão direta com os ataques de **Inundação de Heartbeat** e **Acesso Não Autorizado**, que envolvem 100 clientes simulados gerando um volume massivo de mensagens na camada de aplicação. O SHAP identifica corretamente que a taxa de pacotes e bytes por segundo é o indicador de anomalia mais óbvio. Além disso, a quinta *feature* mais importante é a `flow_max_ocpp16_metervalues_wh_diff` (diferença máxima de energia entre mensagens `meterValues` consecutivas), o que demonstra que o

SHAP também captura a lógica do ataque de **Manipulação do Perfil de Carregamento**, que força um consumo de energia diferente do esperado.

O LIME, em contrapartida, foca em eventos locais e raros. Ele posiciona *flags* TCP incomuns relacionadas a quantidade total de pacotes com *flags* específicas, como `flow_total_URG_flag` (LIME 1º) (pacote urgente), `flow_total_CWR_flag` (LIME 2º) (diminuição da janela de congestionamento) e `flow_total_ECE_flag` (LIME 3º) (conexão congestionada) no topo de seu ranking. Além disso, o LIME identifica sintomas diretos dos ataques de negação de serviço, como o `flow_total_http_5xx_packets` (LIME 4º) (número total de mensagens de erro de servidor HTTP 5XX), que indica a resposta do CSMS sobrecarregado, e o `(flow_total_ocpp16_resetHard_packets)` (LIME 5º) (número total de mensagens OCPP 1.6 Hard Reset da conexão), um comando de aplicação que pode ser usado para fins maliciosos.

Após a análise, sabe-se que o LIME opera gerando explicações ao aproximar o modelo complexo com um modelo substituto local e mais simples, como uma regressão linear [RIBEIRO et al., 2016](#). Ele foca em entender o porquê de uma previsão individual ao analisar o comportamento do modelo na vizinhança da instância de interesse, criada através da perturbação dos dados de entrada. Uma das limitações dessa abordagem é que as explicações podem ser sensíveis à técnica de perturbação utilizada [HERMOSILLA et al., 2025](#).

Por outro lado, o SHAP é fundamentado na teoria dos jogos cooperativos para distribuir de forma justa a previsão do modelo entre os "jogadores", que são as *features* [LUNDBERG e LEE, 2017](#). O método calcula a contribuição média de uma característica em todas as combinações possíveis de *features*, o que lhe confere propriedades teóricas como consistência e completude. Isso permite que o SHAP ofereça explicações mais estáveis, ao capturar interações complexas e fornecer uma visão global coerente [HERMOSILLA et al., 2025](#).

A divergência nos rankings, portanto, é um reflexo direto dessas diferentes filosofias. Uma *feature* como a `Bwd_RST_Flags` pode ter um impacto global que é capturado de forma consistente pela abordagem combinatória do SHAP, posicionando-a no topo da importância.

Contudo, o LIME, ao analisar o modelo através de múltiplas aproximações lineares locais, pode subestimar o impacto dessa mesma *feature* se sua influência for predominantemente não-linear, o que resulta em uma baixa classificação agregada. Em resumo, as diferenças não indicam um erro, mas sim a manifestação da estabilidade teórica e da visão holística do SHAP em contraste com a agilidade e a perspectiva local do LIME [HERMOSILLA et al., 2025](#).

CICFlowMeter

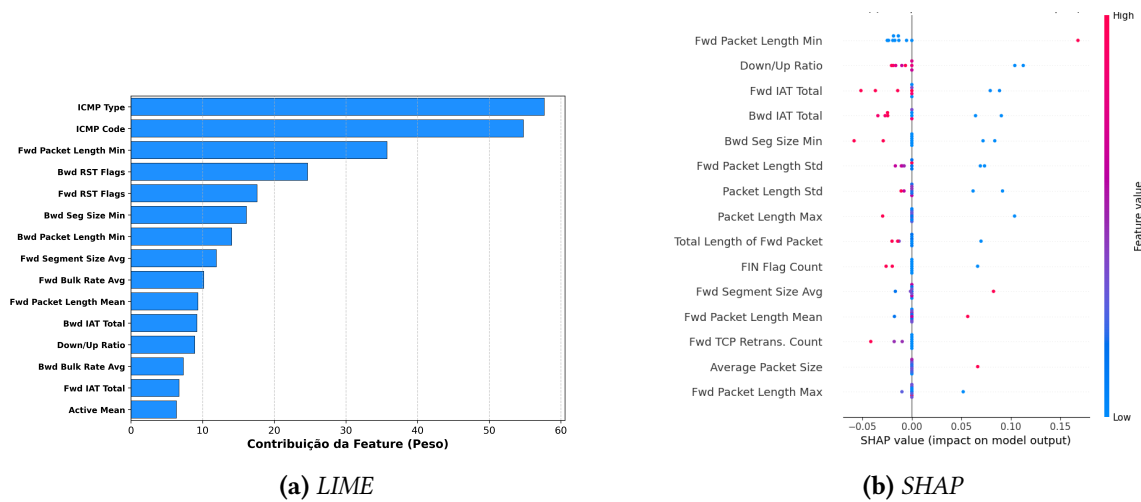


Figura 3.2: Comparação de peso das 15 primeiras *features* entre LIME e o SHAP do CICFlowMeter

A primeira observação, derivada tanto do gráfico de importância do LIME (Figura 3.2a) quanto do SHAP (Figura 3.2b), é que uma porcentagem relativamente pequena das *features* possui uma importância relevante para o resultado final, o que demonstra uma concentração do poder preditivo em um subconjunto pequeno de características. Nestes gráficos, foram selecionadas as 15 *features* mais importantes e suas respectivas contribuições. Na Figura 3.2a, é possível observar que as 2 primeiras possuem uma contribuição muito maior ao comparar com o restante, a partir do Eixo X. Na Figura 3.2b, a partir do Eixo X, que representa o peso de determinadas *features* em uma decisão específica, elas tiveram um poder de decisão muito maior comparada ao restante, algo que é possível de se observar pela quantidade de pontos fora do eixo central. Essa distribuição sugere a presença de características redundantes ou irrelevantes no conjunto de dados original. O uso de métodos de explicabilidade permite quantificar a contribuição de cada *feature* para as previsões do

modelo, tornando possível identificar aquelas com impacto marginal.

Para observar o impacto da remoção das *features* consideradas menos importantes, foram retiradas 8, 16, 32 e 64 entre as menos importantes do *dataset* e feita uma análise de suas métricas. O impacto dessa alteração pode ser observado nos gráficos das Figuras 3.3a e 3.3b, que possuem no Eixo X a quantidade de *features* retiradas e no Eixo Y o valor da métrica, juntamente com a linha pontilhada representando o valor original, sem nenhuma alteração no *dataset*.

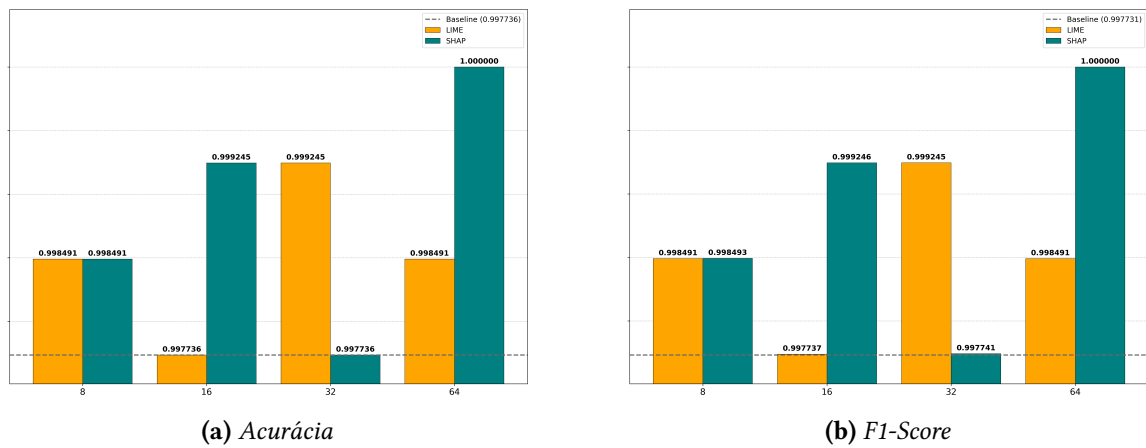


Figura 3.3: Acurácia e F1-Score do dataset CICFlowMeter e suas modificações

A análise dos resultados obtidos após a seleção de características, guiada pelas ferramentas LIME e SHAP, revela um padrão notavelmente positivo. Conforme ilustrado nos gráficos de Acurácia (Figura 3.3a), F1-Score (Figura 3.3b), Recall (Figura 3.4a) e Precisão (Figura 3.4b), a remoção de 8, 16, 32 e até 64 características consideradas menos importantes não apenas preservou o desempenho do modelo de detecção, mas, em diversos casos, levou a uma melhoria mensurável em comparação com o modelo *baseline*, que utilizou o conjunto completo de atributos. O caso mais relevante onde o modelo treinado com as características selecionadas pelo SHAP (no experimento com 64 *features* removidas) atingiu um desempenho perfeito, com Acurácia, Precisão e Recall de 100%. Este resultado valida que a explicabilidade pode ser usada não apenas para interpretar, mas também para otimizar modelos de detecção.

Uma observação notável nos gráficos de desempenho é a grande similaridade — em

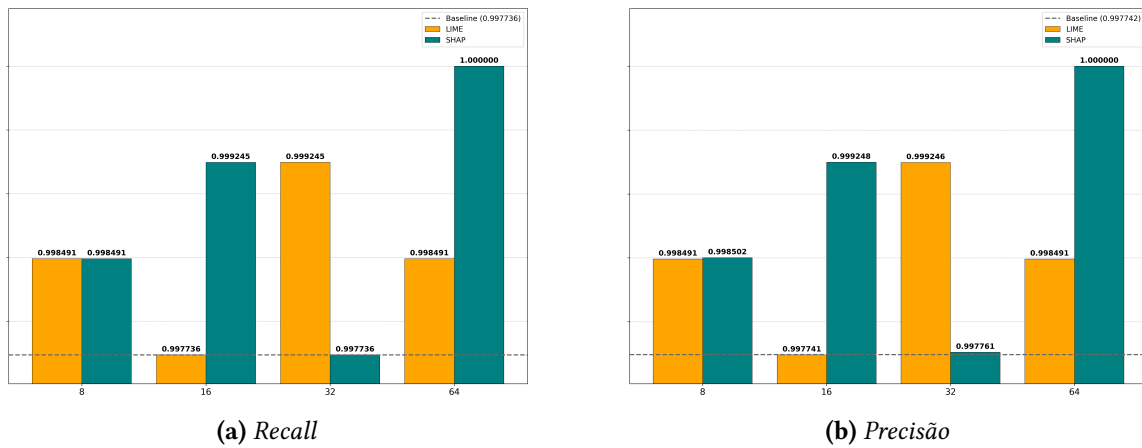


Figura 3.4: Recall e Precisão do dataset CICFlowMeter e suas modificações

muitos casos, a igualdade — entre as métricas de *Acurácia* (Figura 3.3a), *F1-Score* (Figura 3.3b), *Recall* (Figura 3.4a) e *Precisão* (Figura 3.4b). Este fenômeno é uma consequência de duas condições específicas deste experimento. Primeiramente, o *dataset* utilizado foi balanceado, contendo um número igual de amostras para cada classe, conforme descrito na documentação do OCPP-IDS [DALAMAGKAS et al., 2025](#). Em segundo lugar, o modelo demonstrou um desempenho extremamente alto. Quando um modelo treinado em um *dataset* balanceado comete um número de erros muito baixo e quase idêntico de falsos positivos e falsos negativos, as fórmulas matemáticas dessas quatro métricas convergem para um valor comum. Portanto, os resultados idênticos indicam um modelo de alto desempenho que erra de forma simétrica entre as classes.

A melhoria ou manutenção do desempenho, mesmo com um número reduzido de características, é um fenômeno conhecido e pode ser atribuído principalmente à redução de ruído e ao combate ao superajuste (*overfitting*) [GUYON e ELISSEEFF, 2003](#). Conjuntos de dados de alta dimensionalidade, como os de tráfego de rede, contêm características irrelevantes ou redundantes que podem atuar como ruído para uma decisão final, neste caso, de ataque ou não, o que é demonstrado na Figura 3.2. Durante o treinamento, um modelo pode aprender associações coincidentes a partir desse ruído ao invés da verdadeira relação de causa e efeito, o que o torna menos eficaz ao ser exposto a dados novos e não vistos. Ao remover as características de menor importância, o modelo é forçado a construir

suas regras de decisão com base nos sinais mais fortes e preditivos, resultando em um modelo mais simples e com maior capacidade de generalização.

Além das métricas relacionadas ao desempenho para detecção, a abordagem demonstrou um benefício prático significativo: a criação de um modelo mais eficiente. O gráfico da duração do treinamento (Figura 3.5a) se manteve constante, o que pode ser justificado pelo tempo de preparo dos dados e carregamento de bibliotecas do *script* de treinamento ser maior que o cálculo matemático em si. Apesar disso, o gráfico de tamanho do modelo (Figura 3.5b) evidencia que, à medida que mais características são removidas, o tamanho do modelo final em disco diminui consideravelmente. No caso do modelo treinado com 64 características removidas, ele é aproximadamente 22% menor que o modelo *baseline*. Modelos mais leves são mais fáceis de implantar, consomem menos recursos de memória e, crucialmente, tendem a ter um tempo de inferência (o tempo para fazer uma nova predição) menor [GUYON e ELISSEEFF, 2003](#). Essa eficiência é especialmente valiosa em cenários de segurança que exigem respostas rápidas a possíveis tentativas de intrusão ou em ambientes com recursos computacionais limitados, como dispositivos de IoT.

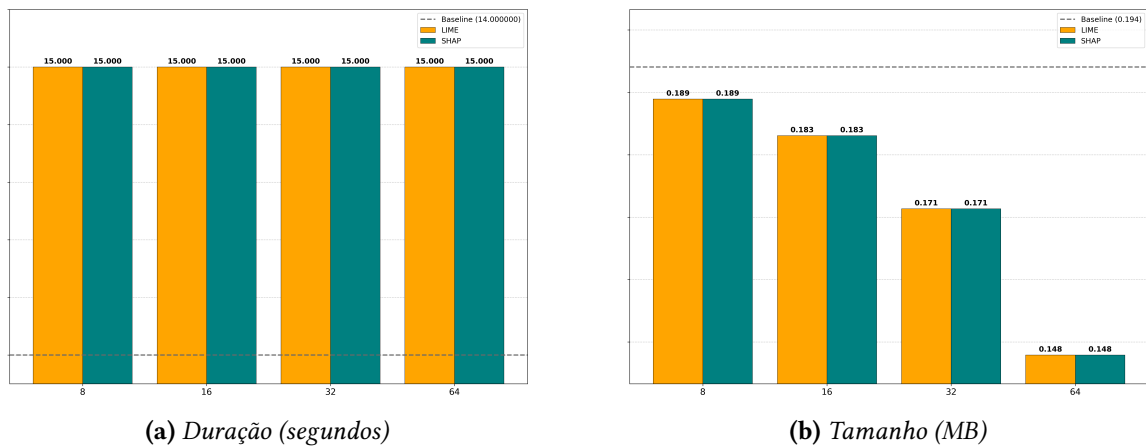


Figura 3.5: Duração do treinamento e tamanho dos modelos do dataset CICFlowMeter e suas modificações

OCPFlowMeter

A análise agora se volta para os experimentos realizados com o *dataset* OCPFlowMeter, que, diferentemente do CICFlowMeter, é composto por características extraídas da camada

de aplicação, específicas do protocolo OCPP. De maneira geral, os resultados obtidos com este *dataset* reforçam a conclusão principal observada anteriormente: a seleção de características guiada por ferramentas de XAI é uma estratégia eficaz para otimizar o desempenho e a eficiência dos modelos de detecção. Pelos mesmos motivos de redução de ruído e combate ao superajuste (*overfitting*) já discutidos, foi possível, em cenários específicos, aprimorar o desempenho do modelo mesmo com um número reduzido de atributos.

Um benefício consistente e replicado nesta segunda rodada de experimentos foi a otimização da eficiência do modelo. Apesar de, novamente, o tempo de treinamento (Figura 3.6a) ter se mantido constante pelos mesmos motivos do CICFlowMeter, o gráfico de tamanho do modelo demonstra (Figura 3.6b), a remoção de 8, 16 e 32 características resultou em uma redução progressiva do tamanho do artefato final, diminuindo de 0.168 MB no modelo *baseline* para até 0.144 MB. Isso confirma que a metodologia proposta produz consistentemente modelos mais leves e práticos, sem impor um custo computacional significativo no treinamento.

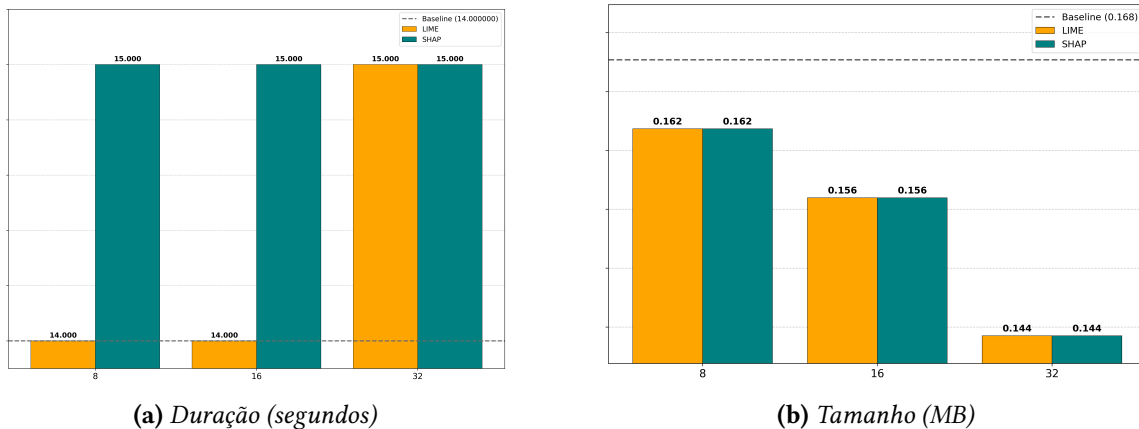


Figura 3.6: Duração do treinamento e tamanho dos modelos do *dataset* OCPPFlowMeter e suas modificações

Contudo, a análise de desempenho para o *dataset* OCPPFlowMeter revela uma interação mais complexa entre as ferramentas LIME e SHAP. Diferentemente dos resultados do CICFlowMeter, onde a abordagem do SHAP frequentemente se mostrou superior, aqui o desempenho ótimo dependeu tanto da ferramenta quanto do número de características removidas. No experimento com 16 características a menos, o modelo otimizado pelo LIME

alcançou um pico de desempenho com Acurácia de 0.9992278 (Figura 3.7a), enquanto a versão do SHAP não superou o *baseline*. Inversamente, ao remover 32 características, foi o modelo guiado pelo SHAP que atingiu o mesmo patamar de excelência, enquanto o do LIME apresentou um desempenho inferior. Nos experimentos com 8 características removidas, ambas as abordagens mantiveram o mesmo nível de desempenho do *baseline*. Ao observar as métricas de Acurácia (Figura 3.7a), *F1-Score* (Figura 3.7b), *Recall* (Figura 3.8a) e *Precisão* (Figura 3.8b), o mesmo comportamento do CICFlowMeter se mantém, em igualdade entre as métricas diferentes de desempenho de detecção do modelo.

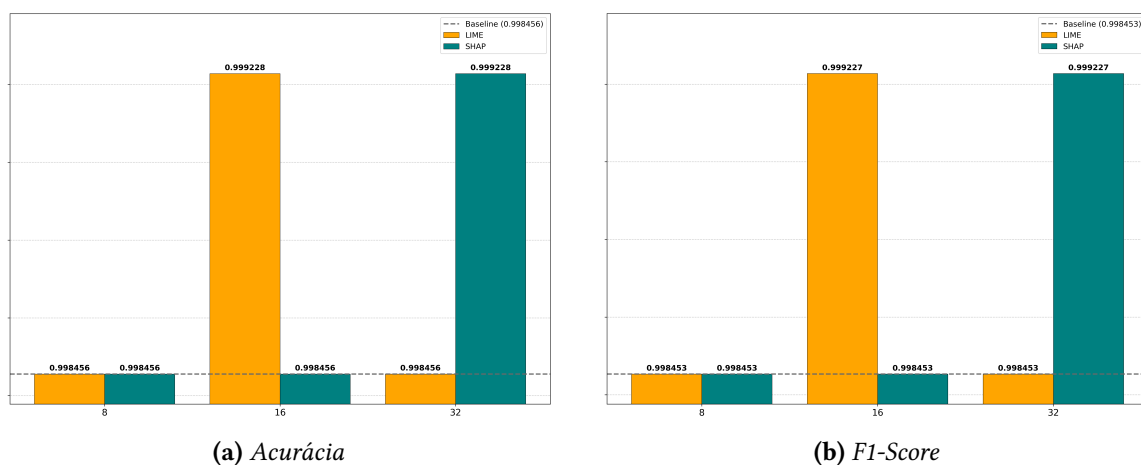


Figura 3.7: Acurácia e F1-Score do dataset OCPPFlowMeter e suas modificações

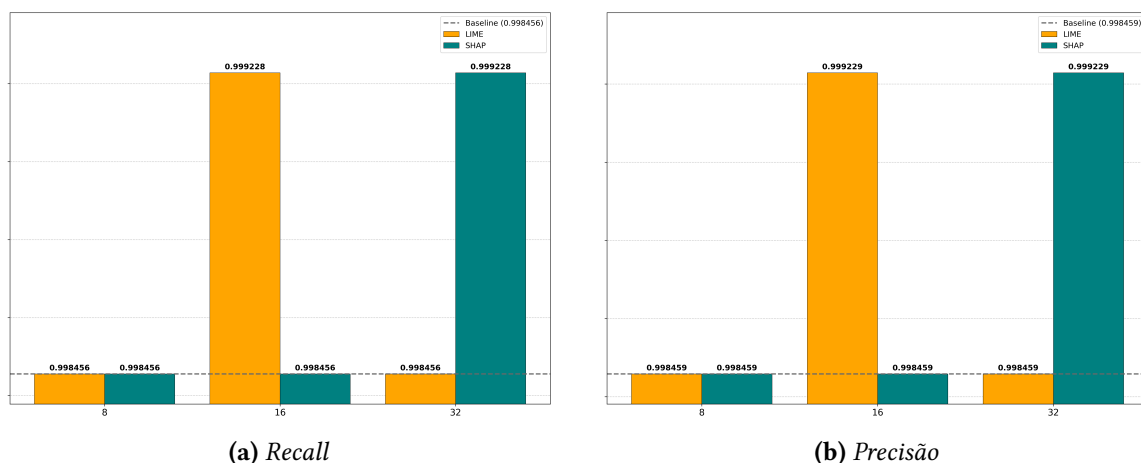


Figura 3.8: Recall e Precisão do dataset OCPPFlowMeter e suas modificações

Adicionalmente, observa-se também no conjunto de Figuras 3.9 que a contribuição individual das características mais importantes tende a ser mais distribuída comparada ao

3.2 | RESULTADOS

dataset anterior, sem que uma única *feature* ou um conjunto de *features* se destaque com um peso desproporcionalmente alto. Isso sugere que a detecção de anomalias na camada de aplicação, específica do protocolo OCPP, é uma tarefa mais complexa, que depende da interação e combinação de múltiplos indicadores sutis. Diferentemente de ataques na camada de rede, os ataques ao OCPP envolvem a manipulação lógica do protocolo. Portanto, o modelo de detecção precisa aprender a partir de um conjunto mais amplo de características que, em conjunto, sinalizam uma atividade maliciosa. Essa distribuição de importância reforça a complexidade do modelo e justifica ainda mais a necessidade de ferramentas de XAI para compreender quais combinações de atributos estão, de fato, influenciando suas decisões.

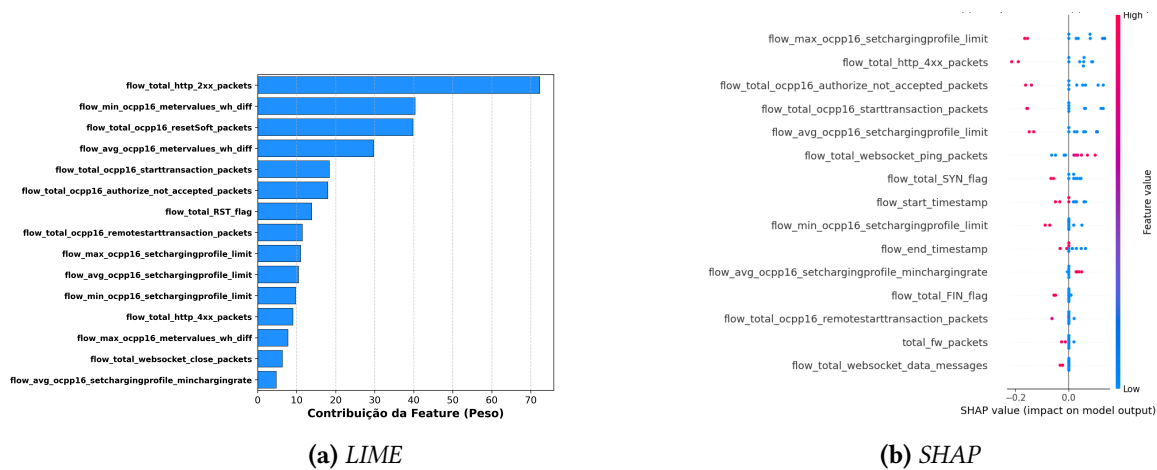


Figura 3.9: Comparação de peso das 15 primeiras *features* entre LIME e o SHAP do OCPPFlowMeter.

Capítulo 4

Conclusões

Este trabalho investigou o desafio da opacidade em modelos de *Machine Learning* aplicados a Sistemas de Detecção de Intrusão. Partindo da premissa de que a maioria dos modelos avançados opera como "caixas opacas", o que compromete a confiança e a validação por parte dos analistas de segurança, o objetivo central foi demonstrar que a Inteligência Artificial Explicável transcende seu papel de mera ferramenta de interpretabilidade e pode se tornar um grande aliado do Analista de Cibersegurança, como uma forma de entender seu ambiente de trabalho e melhorar a confiança em soluções desconhecidas que tomam decisões que podem impactar abruptamente seu ambiente.

A hipótese central era que as ferramentas de XAI, especificamente LIME e SHAP, poderiam ser empregadas como um mecanismo estratégico para otimizar e fortalecer os modelos de detecção. Para validar esta hipótese, foi proposto e aplicado um arcabouço metodológico focado na análise da importância das *features* em um contexto de detecção de ataques cibernéticos, utilizando o modelo empregado em um IDS, chamado OCPP-IDS e seus *datasets* (CICFlowMeter e OCPPFlowMeter) como formas de aplicar esse método em pequena escala.

Os resultados demonstraram que, embora LIME e SHAP apresentem divergências notá-

veis em seus *rankings* de importância, reflexo de suas distintas abordagens teóricas, local e global, respectivamente, ambas as ferramentas forneceram explicações valiosas sobre a lógica dos modelos. A análise revelou que o poder preditivo estava, de fato, concentrado em um subconjunto de características, identificando a presença de uma possível redundância nos conjuntos de dados originais, em que as principais métricas de desempenho foram mantidas ou melhoradas à medida que as métricas de desempenho (tamanho do modelo) melhoraram. Isso é algo muito valioso na situação em que esses modelos necessitam de poder computacional e elétrico equivalente à quantidade de dados carregados, o que pode escalar rapidamente em redes corporativas.

A principal contribuição deste trabalho foi a comprovação de que a otimização baseada em XAI é viável e eficaz. Ao treinar novos modelos utilizando somente as *features* consideradas mais relevantes, foi possível não somente manter, mas em cenários específicos, como o experimento com o CICFlowMeter, melhorar significativamente as métricas de desempenho, atingindo 100% de acurácia, precisão e *recall*. Esta melhoria é atribuída à redução de superajuste (*overfitting*) pela abundância de dados e à focalização do modelo nos indicadores de ameaça mais robustos.

Além da melhoria na detecção, a otimização resultou em benefícios práticos diretos: os modelos otimizados apresentaram uma redução de tamanho considerável, chegando a ser 22% menores que a *baseline*. Modelos mais leves são cruciais para a implantação em ambientes com recursos computacionais limitados, como dispositivos IoT, e permitem um tempo de inferência menor, agilizando a resposta a incidentes, como mostrado em GUYON e ELISSEEFF, 2003.

Este estudo também destacou a complexidade da detecção na camada de aplicação, como visto no *dataset* OCPPFlowMeter, onde a detecção depende de uma combinação mais sutil e distribuída de *features*. Este cenário reforça ainda mais a necessidade de ferramentas XAI para os analistas poderem compreender e validar interações complexas que, de outra forma, permaneceriam ocultas. Além disso, mostra a necessidade de entender seu ambiente

e os protocolos utilizados para tal finalidade de detecção de intrusão.

A explicabilidade não é somente um requisito de confiança ou usabilidade, mas um fator estratégico indispensável para a otimização da eficácia e da robustez dos sistemas de segurança cibernética. O arcabouço proposto permite que especialistas transformem modelos "caixa opaca" em ferramentas mais enxutas, eficientes e, fundamentalmente, mais compreensíveis. Isso pode ser alcançado de forma mais eficiente caso o arcabouço seja seguido desde o início da produção dos scripts responsáveis pela geração dos modelos utilizados nos IDS. Conforme as boas práticas de ciência aberta, todos os códigos e artefatos gerados estão publicamente disponíveis para replicação e futuros trabalhos em <https://github.com/th-duvanel/ids-explainability>.

Como trabalhos futuros, sugere-se a investigação da "outra face" da explicabilidade, mencionada na introdução: a utilização dos rankings de importância de *features* para testar a robustez dos modelos contra ataques adversariais (*adversarial attacks*), que são ataques que exploram informações sobre o modelo e seu *dataset*, que podem ser obtidas justamente a partir da explicabilidade, explorando a troca entre transparência e segurança. Propõe-se também a aplicação de ferramentas de análise e explicabilidade mais robustas, como métodos baseados em gradiente, para validar e complementar os achados do LIME e SHAP, especialmente em modelos de DL. Por fim, sugere-se a aplicação do arcabouço proposto para a criação de um modelo do zero, integrando-o a um pipeline de aprendizado contínuo, para permitir que o sistema trabalhe com novos *datasets* de forma dinâmica, reavaliando e otimizando a seleção de *features* automaticamente à medida que novas ameaças emergem.

Referências

- [ABU-MOSTAFA *et al.* 2012] Yaser S. ABU-MOSTAFA, Malik MAGDON-ISMAIL e Hsuan-Tien LIN. *Learning From Data*. AMLBook, 2012. ISBN: 1600490069 (citado na pg. 9).
- [CAPUANO *et al.* 2022] Nicola CAPUANO, Giuseppe FENZA, Vincenzo LOIA e Claudio STANZIONE. “Explainable artificial intelligence in cybersecurity: a survey”. *IEEE Access* 10 (2022), pp. 93575–93600. DOI: [10.1109/ACCESS.2022.3204171](https://doi.org/10.1109/ACCESS.2022.3204171) (citado na pg. 2).
- [DALAMAGKAS *et al.* 2025] Christos DALAMAGKAS *et al.* *Federated ocpp 1.6 intrusion detection dataset*. 2025. DOI: [10.21227/v1f0-9t13](https://doi.org/10.21227/v1f0-9t13). URL: <https://dx.doi.org/10.21227/v1f0-9t13> (citado nas pgs. 24, 25, 37).
- [DarkTrace s.d.] DarkTrace. <https://www.darktrace.com/>. Acesso em: 26 out. 2025 (citado na pg. 9).
- [FERRAG *et al.* 2021] Mohamed Amine FERRAG, Othmane FRIHA, Leandros MAGLARAS, Helge JANICKE e Lei SHU. “Federated deep learning for cyber security in the internet of things: concepts, applications, and experimental analysis”. *IEEE Access* 9 (2021), pp. 138509–138542. DOI: [10.1109/ACCESS.2021.3118642](https://doi.org/10.1109/ACCESS.2021.3118642) (citado na pg. 28).
- [GUYON e ELISSEEFF 2003] Isabelle GUYON e André ELISSEEFF. “An introduction to variable and feature selection”. *J. Mach. Learn. Res.* 3.null (mar. de 2003), pp. 1157–1182. ISSN: 1532-4435 (citado nas pgs. 37, 38, 44).

- [HERMOSILLA *et al.* 2025] Pamela HERMOSILLA, Sebastián BERRÍOS e Héctor ALLENDE-CID. “Explainable ai for forensic analysis: a comparative study of shap and lime in intrusion detection models”. *Applied Sciences* 15.13 (2025). ISSN: 2076-3417. DOI: [10.3390/app15137329](https://doi.org/10.3390/app15137329). URL: <https://www.mdpi.com/2076-3417/15/13/7329> (citado nas pgs. 34, 35).
- [KHRAISAT *et al.* 2019] Ansam KHRAISAT, Iqbal GONDAL, Peter VAMPLEW e Joarder KAMRUZZAMAN. “Survey of intrusion detection systems: techniques, datasets and challenges”. *Cybersecurity* 2 (dez. de 2019). DOI: [10.1186/s42400-019-0038-7](https://doi.org/10.1186/s42400-019-0038-7) (citado na pg. 7).
- [LU *et al.* 2020] Yunlong LU, Xiaohong HUANG, Yueyue DAI, Sabita MAHARJAN e Yan ZHANG. “Blockchain and federated learning for privacy-preserved data sharing in industrial iot”. *IEEE Transactions on Industrial Informatics* 16.6 (2020), pp. 4177–4186. DOI: [10.1109/TII.2019.2942190](https://doi.org/10.1109/TII.2019.2942190) (citado na pg. 28).
- [LUNDBERG e LEE 2017] Scott M. LUNDBERG e Su-In LEE. “A unified approach to interpreting model predictions”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS’17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 4768–4777. ISBN: 9781510860964 (citado nas pgs. 13, 34).
- [NEUPANE *et al.* 2022] Subash NEUPANE *et al.* “Explainable intrusion detection systems (x-ids): a survey of current methods, challenges, and opportunities”. *IEEE Access* 10 (2022), pp. 112392–112415. DOI: [10.1109/ACCESS.2022.3216617](https://doi.org/10.1109/ACCESS.2022.3216617) (citado nas pgs. 1, 2, 12, 15–17, 19–21).

REFERÊNCIAS

- [RIBEIRO *et al.* 2016] Marco Tulio RIBEIRO, Sameer SINGH e Carlos GUESTRIN. “"why should i trust you?": explaining the predictions of any classifier”. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: Association for Computing Machinery, 2016, pp. 1135–1144. ISBN: 9781450342322. DOI: [10.1145/2939672.2939778](https://doi.org/10.1145/2939672.2939778). URL: <https://doi.org/10.1145/2939672.2939778> (citado nas pgs. 12, 34).
- [SHAP (SHapley Additive exPlanations) s.d.] SHAP (SHapley Additive exPlanations). <https://shap.readthedocs.io>. Acesso em: 04 out. 2025 (citado na pg. 19).
- [STALLINGS 2013] William STALLINGS. *Cryptography and Network Security: Principles and Practice*. 6th. USA: Prentice Hall Press, 2013. ISBN: 0133354695 (citado na pg. 6).
- [STALLINGS e BROWN 2014] William STALLINGS e Lawrie BROWN. *Computer Security: Principles and Practice*. 3rd. USA: Prentice Hall Press, 2014. ISBN: 0133773922 (citado na pg. 6).