Uma Arquitetura baseada em Inteligência Artificial Explicável (XAI) para Sistemas de Detecção de Intrusões em Smart Grids

Camilla Borchhardt Quincozes[1], Henrique C. Oliveira[2], Silvio E. Quincozes[1], Rodrigo S. Miani[2], Vagner E. Quincozes[3]

[1]Campus Alegrete – Universidade Federal do Pampa (UNIPAMPA), Brasil

[2]FACOM – Universidade Federal de Uberlândia (UFU), Brasil

[3]Instituto de Computação (IC) – Universidade Federal Fluminense (UFF) – Brasil.

camillaborchhardt.aluno@unipampa.edu.br, henrique.cr@ufu.br

silvioquincozes@unipampa.edu.br, miani@ufu.br, vequincozes@id.uff.br

Abstract. *This paper proposes an architecture for an Explainable Intrusion Detection System (X-IDS) for electrical substations, aiming to enhance the transparency and reliability of traditional IDSs. The architecture integrates eXplainable Artificial Intelligence techniques (XAI) and new feature extraction methods, using temporal enrichment and robust preprocessing to improve the detection and interpretation of attacks. The results demonstrate that the proposed X-IDS reduces bias towards certain attacks, enhances the interpretation of complex attacks, and facilitates the analysis of corrections and new implementations, offering a more robust and transparent solution for the security of electrical substations. Random Forest presented the best performance statistics: accuracy and precision of 98.79%, and recall 98.68%.*

Resumo. *Este trabalho propõe uma arquitetura de Sistema de Detecção de Intrusões Explicável (X-IDS) para subestações elétricas, visando aumentar a transparência e confiabilidade dos IDSs tradicionais. A arquitetura integra técnicas de Inteligência Artificial Explicável (XAI) e novos métodos de extração de atributos, utilizando enriquecimento temporal e pré-processamento robusto para melhorar a detecção e interpretação de ataques. Os resultados demonstram que o X-IDS proposto reduz o viés para certos ataques, aprimora a interpretação de ataques complexos e facilita a análise de correções e novas implementações, oferecendo uma solução mais robusta e transparente para a segurança das subestações elétricas. Random Forest apresentou as melhores métricas de desempenho: acurácia e precisão de 98,79%, e revocação 98,68%.*

1. Introdução

A *Inteligência Artificial* (IA) é uma tecnologia revolucionária que tem demonstrado impressionantes habilidades para diversas aplicações [Davenport 2018]. No setor elétrico, onde há um crescente volume de ameaças cibernéticas [Youssef et al. 2016], a IA surge como uma alternativa para fornecer uma resposta através do seu emprego na implementação de Sistemas de Detecção de Intrusão (IDS) [Neupane et al. 2022].

Segundo [Molnar 2022], é crucial que as decisões tomadas por IDSs possam ser auditadas e compreendidas tanto por especialistas quanto por reguladores. No entanto, há

sérios desafios, especialmente no que diz respeito à transparência e explicabilidade das decisões tomadas por IDSs baseados em IA. Tipicamente, as entradas e saídas são conhecidas, mas os processos decisórios internos permanecem obscuros. A necessidade de IDSs não apenas eficientes, mas também interpretáveis e transparentes destaca a importância da *Inteligência Artificial Explicável* (XAI) [Quincozes et al. 2024].

Na literatura, há alguns poucos esforços no emprego de estratégias explicáveis para o setor elétrico [Kuzlu et al. 2020]. Ademais, os trabalhos existentes carecem da real explicabilidade, a qual requer a combinação de ferramentas XAI com informações do domínio da aplicação em que o IDS está implantado. Portanto, este trabalho busca preencher tais lacunas, explorando alternativas para a construção de um X-IDS (ou IDS Explicável) que combina a eficiência com transparência e interpretabilidade. A adoção de XAI nos IDS oferece uma oportunidade para não só aprimorar a segurança de infraestruturas críticas, mas também avançar no desenvolvimento de tecnologias de IA que são mais transparentes e confiáveis.

O principal objetivo deste trabalho consiste na implementação de um X-IDS, onde três algoritmos populares são avaliados: *Random Forest*, *Decision Tree* e *XGBoost*. Para cada algoritmo, experimentos envolvendo seis tipos de ataques direcionados a subestações elétricas são conduzidos e os resultados são analisados à luz da explicabilidade por meio da ferramenta de XAI chamada SHAP. Tais ataques são gerados pela ferramenta ERENO [Quincozes et al. 2023]. Como contribuição adicional, são propostos oito novos atributos para a melhoria do desempenho do IDS, os quais também são analisados por meio de ferramentas de explicabilidade.

O restante deste trabalho está estruturado da seguinte forma: A Seção 2 discute os conceitos chave relacionados a IDS, XAI, subestações elétricas, a norma IEC-61850, e o processo de Engenharia de Atributos (do inglês, *Feature Engineering*). A Seção 3 descreve os trabalhos relacionados. A Seção 4 descreve os métodos utilizados e a investigação do problema de pesquisa. Então, a Seção 5 apresenta a técnica de enriquecimento de atributos adotadas. Em seguida, a Seção 6 apresenta os experimentos realizados, os resultados obtidos e a discussão desses resultados. Por fim, a Seção 7 apresenta as contribuições do trabalho e pesquisas futuras.

2. Detecção de intrusão e explicabilidade em subestações elétricas

Subestações elétricas são componentes críticos na distribuição de energia elétrica, funcionando como pontos de interseção que transformam e distribuem eletricidade de maneira eficiente e segura. A norma IEC-61850 foi desenvolvida para padronizar a comunicação dentro dessas subestações, garantindo interoperabilidade entre dispositivos de diferentes fabricantes [IEC 2003]. Protocolos como o *GOOSE* e *SV*, definidos pela IEC-61850, permitem a troca confiável de mensagens de controle e status, facilitando a implementação de funções automatizadas de proteção e controle [Quincozes et al. 2021].

Com o aumento das ameaças cibernéticas, a necessidade de IDSs em subestações elétricas tornou-se evidente. IDS são projetados para identificar atividades anômalas que possam indicar tentativas de intrusão. Recentemente, houve um avanço significativo na aplicação de Aprendizado de Máquina (Machine Learning – ML) em subestações elétricas, melhorando a capacidade de detecção e resposta a incidentes de segurança [Premaratne et al. 2010].

No entanto, a complexidade dos modelos de ML e a natureza "caixa preta" de muitos desses sistemas criam desafios em termos de transparência e interpretabilidade. Aqui, os Sistemas de Detecção de Intrusão Explicáveis (X-IDS) se tornam necessários. X-IDS utilizam técnicas de XAI para fornecer *insights* claros sobre as decisões dos modelos de ML. XAI é uma subárea complementar à IA que visa tornar os modelos de ML mais transparentes e compreensíveis para humanos, utilizando métodos como *SHAP* (*SHapley Additive exPlanations*) [Lundberg and Lee 2017] e *LIME* (*Local Interpretable Model-agnostic Explanations*) [Ribeiro et al. 2016] para explicar como os atributos individuais influenciam as previsões dos modelos [Vainio-Pekka et al. 2023].

Em particular, um conjunto de técnicas de ML úteis para implementar IDSs, inclusive em subestações elétricas, são os classificadores. Eles viabilizam a detecção de intrusões, tipicamente, por meio do aprendizado supervisionado, onde o algoritmo constrói um modelo de predição através de assinaturas rotuladas [Suaboot et al. 2020]. As principais famílias de classificadores incluem árvores de decisão, florestas aleatórias, máquinas de vetores de suporte e métodos de *boosting*. Alguns dos classificadores facilitam a explicabilidade intrinsecamente, mas nem todos contemplam essa possibilidade [Bisong and Bisong 2019]:

- *Decision Tree*: Naturalmente explicável. Ele fornece uma estrutura intuitiva de tomada de decisão com base em regras simples derivadas dos dados.
- *Random Forest*: Não é naturalmente explicável. Combina múltiplas árvores de decisão para melhorar a precisão, mas a combinação das árvores dificulta a interpretação.
- *XGBoost*: Não é naturalmente explicável. É um método de *boosting* que aumenta a precisão por meio de árvores de decisão sequenciais, mas a complexidade das combinações torna a interpretação desafiadora.

Em geral, esses classificadores requerem técnicas de explicabilidade porque, embora eficazes, sua complexidade impede que os operadores entendam facilmente como as decisões são tomadas. A falta de transparência pode dificultar a detecção de falhas e a implementação de melhorias. Mesmo o *Decision Tree* pode ser beneficiado com técnicas que ampliam a capacidade de interpretabilidade de árvores de decisões mais complexas.

Técnicas de XAI visam tornar os modelos de ML mais transparentes e compreensíveis. Métodos como SHAP e LIME são fundamentais para explicar o comportamento dos modelos. A biblioteca SHAP atribui valores a cada atributo individual, mostrando sua contribuição para a previsão do modelo [Lundberg and Lee 2017]. Já a biblioteca LIME, cria modelos locais simples para explicar previsões individuais, tornando mais fácil entender como o modelo global opera em casos específicos [Ribeiro et al. 2016]. A aplicação dessas técnicas é crucial para garantir que os IDS sejam não apenas eficazes, mas também interpretáveis e confiáveis para os operadores humanos. A transparência fornecida pela XAI permite uma melhor análise e compreensão das decisões do modelo, facilitando a implementação de melhorias e a detecção de falhas.

3. Trabalhos Relacionados

A Tabela 1 resume os trabalhos mais relacionados à aplicação de técnicas de XAI em diversos domínios, detalhando os datasets utilizados, classificadores aplicados e os tipos de ataques abordados. Em seguida, cada um desses trabalhos é descrito.

Tabela 1. Comparação dos Trabalhos Relacionados

Referência	Domínio	Técnicas XAI	Datasets	Classificadores	Ataques
[Wang et al. 2020]	Redes Corporativas	SHAP	NSL-KDD	*One-vs-All*, *Multiclass*	DoS, U2R, R2L, Probe
[Sivamohan et al. 2023]	Sistemas Ciberfísicos	SHAP	NSL-KDD, CIC-IDS	LSTM, GRU, BiLSTM, TEA-EKHO-IDS	DoS, Probe, R2L, U2R, Bot Web, *Bruteforce*, *Infiltration*, *Injection*
[Kuzlu et al. 2020]	Geração Fotovoltaica	SHAP, LIME, ELi5	GEFCOM	RFR	Não se aplica
[Munir et al. 2023]	Recursos Energéticos Distribuídos	SHAP	SCADA, WUSTL-IIOT-2018	*Random Forest*, *Extra Tree*, *Gradient Boosting*, *AdaBoost*, *Linear Regression*	*PortScanner*, *AdressScan*, *Device Id.Atack*, *Agressive Mode*, *Exploit*
[Zolanvari et al. 2021]	IIoT	TRUST LIME	SCADA, NSL-KDD, WUSTL-IIOT-2018, UNSW	ANN	*Backdoor*, *Injection*, DoS, *Reconnaissance*
[Dresch et al. 2024]	Redes Intra-veiculares	SHAP	X-CAN-IDS	XGBoost	Fabricação *Fuzzing*
Este trabalho	**Subestações Elétricas**	**SHAP**	**ERENO**	***XGBoost*, *Random Forest*, *Decision Tree***	*High StNum*, *Injection*, *Inverse Replay*, *Masquerade*, *Poisoned High Rate*, *Random Replay*

A aplicação de técnicas de XAI em IDS é explorada em diversos estudos. Por exemplo, os autores [Wang et al. 2020] propõem um *framework* utilizando SHAP para melhorar a transparência dos IDSs baseados em ML. O estudo destaca a comparação entre explicações locais e globais, utilizando o dataset NSL-KDD. As explicações locais fornecem as razões por trás das decisões do modelo em entradas específicas, enquanto as explicações globais identificam os recursos importantes e suas relações com diferentes tipos de ataques. Contudo, a pesquisa aborda a necessidade de otimizar a eficiência computacional do SHAP em ambientes de grande escala.

Complementando essa abordagem, [Sivamohan et al. 2023] introduzem o TEA-EKHO-IDS, um sistema que combina XAI e Otimização de Cardume de Krill Aprimorada, do inglês, *Enhanced Krill Herd Optimization Algorithm* (EKHO) para detectar e classificar intrusões em sistemas ciberfísicos industriais. Utilizando XAI-EKHO para seleção de características e integrando LSTM bidirecional com Otimização Bayesiana (BO-Bi-LSTM), o sistema demonstra uma acurácia de 98,96%. No entanto, a complexidade do método sugere a necessidade de investigações adicionais para acelerar o processo.

No domínio da previsão de geração de energia, [Kuzlu et al. 2020] exploram o uso de técnicas de XAI, como LIME, SHAP e ELI5, para melhorar a previsibilidade e transparência dos modelos de previsão de energia solar fotovoltaica (PV). O estudo destaca a aplicação de XAI para destacar a influência de variáveis específicas nas previsões do modelo. Todavia, uma limitação consiste na utilização de dados em tempo real para aumentar a adaptabilidade e precisão dos modelos preditivos, aumentando a dependência por dados históricos de alta qualidade.

O desenvolvimento de um *framework* de IA confiável para detectar e explicar a causa raiz de ataques cibernéticos em recursos energéticos distribuídos foi realizado por

[Munir et al. 2023]. Integrando métodos de ML com XAI, a pesquisa visa criar sistemas mais transparentes e justificáveis. A potencial redução na precisão do modelo ao priorizar a explicabilidade é uma limitação destacada, sugerindo a necessidade de equilibrar esses dois aspectos em futuros estudos.

O aumento da segurança em ambientes da Internet das Coisas Industrial (IIoT) utilizando uma combinação de técnicas de XAI, TRUST e LIME foi explorado por [Zolanvari et al. 2021]. Este trabalho demonstra como as técnicas de XAI podem ser implementadas em sistemas industriais para melhorar a segurança e a confiança. A principal limitação reside na escalabilidade das técnicas de XAI, indicando que pesquisas futuras devem focar em métodos menos onerosos computacionalmente.

Em redes veiculares, SHAP foi usado junto ao classificador XGBoost por [Dresch et al. 2024] para fornecer explicabilidade aos ataques de fabricação e *fuzzing*. No entanto, os ataques e o cenário estudado no trabalho não cobre os cenários e ataques deste trabalho.

Embora esses trabalhos mostrem a eficácia das técnicas de XAI em diversos domínios de sistemas de detecção de intrusões, poucos investigam sua aplicação em subestações elétricas. Além disso, este estudo é o único que emprega técnicas de enriquecimento temporal com o *dataset* ERENO [Quincozes et al. 2023] para aprimorar tanto a precisão das previsões quanto a qualidade dos dados. Estas contribuições diferenciam significativamente nosso trabalho, ressaltando seu valor único para o campo de *smart grids*, particularmente, em subestações elétricas.

4. Desenvolvimento de X-IDSs para Subestações Elétricas

Nesta seção, são apresentados os componentes da arquitetura proposta para o desenvolvimento de um *IDS Explicável* (X-IDS). Tais componentes são sumarizados na Figura 1. O código-fonte desenvolvido está disponível publicamente[1].

Figura 1. Arquitetura X-IDS proposta.

[1]Código-fonte: `https://github.com/Henriqw3/tcc-ereno-xai-ids`

O processo começa com a inserção das entradas na interface *web* do gerador de tráfego, incluindo parâmetros de tráfego da subestação elétrica, modelos de ataques e *uploads* de sinais elétricos (corrente e tensão) reais ou realistas. Em seguida, tais informações são enviadas para o núcleo do gerador de tráfego [Quincozes et al. 2023], o qual envolve a simulação de mensagens GOOSE e SV, seguindo o padrão IEC-61850, além da execução de comportamentos maliciosos (ataques) ao protocolo GOOSE. Além da classe nromal, são carregados sete tipos de ataques, conforme listados a seguir:

- *high StNum*: Explora a configuração de valores altos e inconsistentes para o número de *status* (*i.e.*, `StNum`), com valores de 10.000 a 100.000, tornando os valores legítimos obsoletos e descartados pelos dispositivos subscritores.
- *injection*: Ataques de injeção aleatória onde o invasor fabrica e transmite mensagens falsas com modificações aleatórias, sem observar padrões de tráfego de rede, mas sem violar os padrões IEC-61850.
- *inverse replay*: Repetição de mensagens com status inverso, causando operações indesejadas como fechar disjuntores durante eventos de falha, potencialmente danificando equipamentos ou colocando vidas em risco.
- *masquerade fake fault*: Transmissão de mensagens GOOSE falsas que imitam eventos de falha em situações normais, visando causar interrupções no fornecimento de energia.
- *masquerade fake normal*: Transmissão de mensagens GOOSE falsas que imitam eventos de situação normal em situações de faltas, visando causar reestabelecimento indevido no fornecimento de energia.
- *poisoned High Rate*: Ataque de inundação de alta taxa, onde o invasor envia múltiplas mensagens falsas por segundo, dificultando a distinção entre mensagens legítimas e falsas.
- *random replay*: Captura e retransmissão aleatória de mensagens GOOSE previamente enviadas, podendo causar diversas consequências negativas dependendo do conteúdo e contexto das mensagens retransmitidas.

Dentro do processo de geração de tráfego, há a engenharia de atributos que resulta em um conjunto de dados. A aplicação de técnicas de extração de novos atributos pela ideia de extrair estatísticas de segmentos de janelas móveis de tempo não sobrepostas foi levada em consideração após a investigação inicial das métricas e matrizes de confusão das classificações dos ataques. O conjunto de dados é utilizado pelo módulo de detecção de intrusões para o treinamento e validação dos algoritmos *XGBoost* e *Decision Tree*. A escolha desses algoritmos foi motivada por sua popularidade na literatura, além de seus resultados em experimentos preliminares terem revelado que são algoritmos competitivos.

Por fim, o módulo de Explicabilidade é uma extensão do módulo IDS, tornando-o um X-IDS. O X-IDS usa a biblioteca SHAP para a explicabilidade dos modelos gerados pelos classificadores do IDS, e os resultados finais são transformados em gráficos que são retornados ao usuário final. Tais gráficos incluem tanto as métricas de desempenho dos classificadores — onde são consideradas métricas como a precisão (*precision*), a sensibilidade (revocação) e a pontuação F1 (*F1-Score*) — quanto os gráficos gerados pelas ferramentas de XAI.

Para aprimorar o modelo, foi usada a função *calculate shap parallel*, que paralelizou o cálculo dos valores SHAP para acelerar o processo. Isso foi necessário devido

ao grande número de instâncias no conjunto de teste. Uma vez obtidos os valores SHAP, foram gerados vários gráficos para visualizar e interpretar as explicações do modelo, incluindo gráficos de dependência para entender como uma variável afeta as previsões do modelo em relação a outra, bem como gráficos de resumo global para identificar quais variáveis são mais importantes para o modelo.

5. Enriquecimento de Atributos

Uma vez que a arquitetura foi estabelecida, um método adicional foi implementado a fim de experimentar-se novos atributos. Tal método se baseou na extração de atributos por enriquecimento de dados de tempo. Essa técnica consiste em integrar o conjunto de dados a uma linha do tempo, de modo a facilitar a detecção de determinados tipos de ataques que possuem relações com séries temporais.

Particularmente, adotou-se uma abordagem baseada na técnica de *Non-Overlapping Moving Window*, que divide os dados em intervalos de tempo fixos e não sobrepostos, realizando análises específicas dentro de cada intervalo. Desta forma, o conjunto de dados é segmentado em uma série temporal, na qual, neste trabalho, optou-se por intervalos heurísticos de 2 segundos. Isso permite realizar uma ou várias operações específicas dentro de cada intervalo sem sobreposição entre eles, destacando as tendências temporais relevantes.

Para melhor compreensão, pode-se elaborar matematicamente a técnica, definindo o procedimento que segmenta uma série temporal y_t em intervalos fixos de tempo de 2 segundos, sem sobreposição entre os intervalos. Cada intervalo é denotado como I_k, onde k é o índice desse intervalo. A ideia consiste em agrupar os valores da série temporal em blocos de tempo fixos e aplicar operações ou análises dentro de cada intervalo separado. Eis uma descrição matemática da abordagem, sendo $t_1, t_2, t_3, \ldots, t_n$ uma sequência de tempos em que os dados são registrados. Há a possibilidade de definir os intervalos de tempo fixos de 2 segundos como:

$$I_k = [t_{2k}, t_{2k+1}] \quad \texttt{para} \quad k = 0, 1, 2, \ldots$$

Onde:

- Cada I_k é um intervalo de tempo de 2 segundos;
- Para cada intervalo I_k, agrupa-se os valores da série temporal y_t que estão dentro desse intervalo;
- Pode-se denotar os valores da série temporal dentro do intervalo I_k como y_{t_j} para $t_j \in I_k$.

Uma vez que os valores da série temporal dentro de I_k estão identificados (ou seja, y_{t_j} para $t_j \in I_k$), pode-se aplicar qualquer análise ou operação desejada dentro desse intervalo, sendo $|I_k|$ o número de pontos de dados dentro de I_k. Por exemplo, para calcular a média dos valores dentro de I_k, é possível aplicar:

$$\mathrm{f}(I_k) = \frac{1}{|I_k|} \sum_{t_j \in I_k} y_{t_j}$$

Desta forma, além dos atributos originais reportados em [Quincozes et al. 2023], oito novos atributos foram extraídos usando enriquecimento por janela de tempo:

média (`window_mean_time`), variância (`window_variance_time`), desvio padrão (`window_std_deviation`), *kurtosis* (`window_kurtosis`), *skewness* (`window_skewness`), número de mensagens da janela (`window_size`), além dos valores mínimos (`window_min_timestamp`) e máximos (`window_max_timestamp`) dos carimbos de tempo das amostras dentro de cada janela.

A *kurtosis* mede o achatamento da distribuição dos tempos em relação à média, indicando o quanto os dados estão concentrados; uma *kurtosis* alta sugere uma distribuição com caudas pesadas e um pico mais proeminente, enquanto uma *kurtosis* baixa indica um pico mais achatado, sugerindo uma variação menos extrema dos dados ao redor da média.

Por outro lado, a *skewness* determina a assimetria da distribuição em relação à média: uma *skewness* positiva implica que a cauda da distribuição se estende para a direita, indicando uma distribuição com inclinação positiva, enquanto uma *skewness* negativa mostra que a cauda se estende para a esquerda, caracterizando uma distribuição com inclinação negativa. Por fim, um atributo chamado *dt_clock* foi criado a partir do atributo existente *GooseTimestamp*, permitindo uma maior granularidade para essa informação.

6. Avaliação

Os resultados da avaliação do X-IDS implementado são apresentados a seguir. Esta seção está organizada com base nos algoritmos classificadores utilizados, sendo que para cada algoritmo será apresentado o seguinte: Primeiramente, são apresentadas as métricas de desempenho na detecção de intrusões. Em seguida, são apresentadas as Matrizes de Confusão. Por fim, são demonstrados os resultados em termos de explicabilidade dos modelos de cada um dos classificadores avaliados no cenário estudado.

Para os experimentos executados, foram geradas 1.000 amostras para cada classe de ataque (totalizando 5.000 amostras de ataques) e outras 5.000 amostras normais, a fim de manter-se um balanceamento entre ataques e amostras normais. O conjunto de dados resultante foi dividido em duas porções, sendo 80% para treino e 20% para teste. Os resultados são sumarizados na Figura 2.

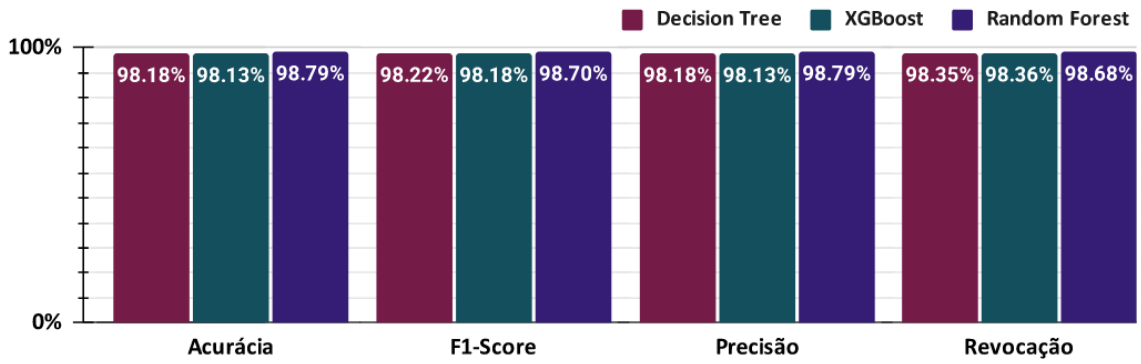


Figura 2. Resultados comparativos do desempenho dos classificadores.

O classificador *Decision Tree* alcançou uma acurácia de 98,18%, precisão de 98,18 e um F1-Score de 98,22%. Esses resultados indicam um desempenho muito consistente, com a capacidade do modelo de identificar corretamente as instâncias positivas (revocação) sendo ligeiramente superior à sua precisão. A proximidade entre as métricas

sugere que o modelo está bem balanceado entre evitar falsos positivos e falsos negativos, resultando em um F1-Score elevado, que é uma média harmônica da precisão e da revocação.

Já para o classificador *XGBoost*, houve uma ligeira queda no F1-Score em comparação com o classificador *Decision Tree*. Essa queda é um reflexo da queda na precisão, mesmo que haja um aumento modesto na revocação. No entanto, tais resultados continuam a indicar um desempenho consistente, com a capacidade do modelo de identificar corretamente as instâncias positivas sendo também ligeiramente superior à sua precisão. Em particular, o classificador *XGBoost* alcançou uma precisão de 98,13%, um revocação de 98,36%, resultando em um F1-Score de 98,18%. Assim como no modelo do classificador *Decision Tree*, a proximidade entre as métricas sugere que o modelo está bem balanceado entre evitar falsos positivos e falsos negativos, resultando em um F1-Score relativamente alto.

Por fim, o classificador *Random Forest* apresentou as melhores métricas, superando os demais algoritmos testados. Sua acurácia e precisão alcançaram 98,79%, o que representa os maiores valores em comparação com os demais classificadores. A revocação desse algoritmo, embora tenha sido a métrica com o valor mais baixo para ele, também se destacou em comparação com a revocação dos outros algoritmos, atingindo 98,68%. Como resultado, a F1-Score do *Random Forest* apresentou uma leve queda de 0,7% em relação à acurácia e precisão, mas ainda assim manteve-se no nível mais elevado.

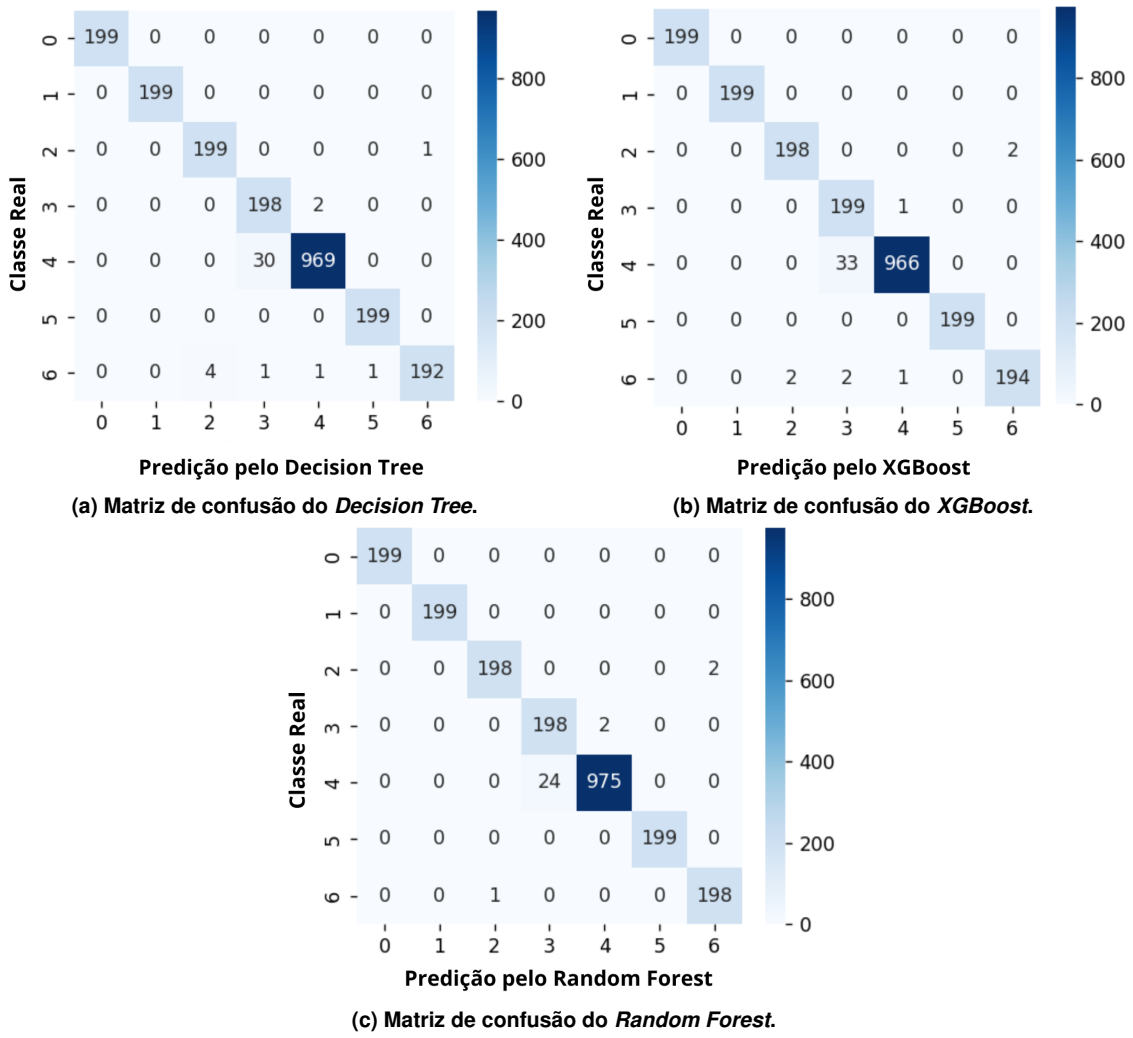
A partir dos resultados apresentados, é possível dois tipos de investigação: i) quais foram as situações em que o classificador confundiu uma amostra de determinada classe, classificando-a como uma classe diferente; e ii) quais foram as informações que explicam as decisões tomadas. A primeira investigação é realizada neste trabalho por meio da análise das matrizes de confusão (Seção 6.1), na próxima seção. Em seguida, a segunda questão é investigada com o uso de técnicas de XAI (Seção 6.2).

6.1. Matrizes de Confusão

A seguir, são apresentadas e comparadas as matrizes de confusão referentes aos resultados obtidos pelos algoritmos *Decision Tree*, *XGBoost* e *Random Forest*.

A matriz de confusão, ilustrada na Figura 3a, detalha a performance do classificador *Decision Tree*. Cada linha representa as verdadeiras classes, e cada coluna representa as predições do modelo. Assim, a diagonal principal concentra os acertos — quando uma classe predita corresponde à classe verdadeira. A matriz revela que o modelo tem uma excelente performance na maioria das classes, com a maioria dos valores concentrados na diagonal principal, indicando predições corretas.

Notavelmente, a classe "normal" (4) tem 969 predições corretas, mas também apresenta 30 falsos negativos, onde ataques foram classificados como normais. As classes de ataques como *high StNum* (0), *injection* (1), *inverse replay* (2), *masquerade fake fault* (3) e *poisoned high rate* (5) mostram que o classificador conseguiu identificar quase todas as instâncias corretamente, com apenas alguns erros esporádicos. A classe *random replay* (6) apresenta mais erros, sendo 4 das amostras desta classe classificadas como *inverse replay* e outras três amostras foram classificadas como *masquerade fake fault*, *normal* e *poisoned high rate*.



(a) Matriz de confusão do *Decision Tree*.

(b) Matriz de confusão do *XGBoost*.

(c) Matriz de confusão do *Random Forest*.

Figura 3. Comparação das Matrizes de Confusão.

Esses resultados demostram que, embora o *Decision Tree* tenha alto desempenho geral, ele tem uma leve dificuldade em distinguir entre a classe "normal" e alguns tipos específicos de ataques, especialmente *random replay*. Melhorias adicionais podem focar em técnicas para reduzir esses erros específicos, possivelmente através de um ajuste mais fino dos hiperparâmetros ou da integração de métodos de ensemble para aumentar a robustez do modelo.

A matriz de confusão para o *XGBoost*, ilustrada na Figura 3b, apresenta um desempenho que é comparável ao do classificador *Decision Tree*. Assim como o *Decision Tree*, o *XGBoost* também mostra a maioria das predições corretas concentradas na diagonal principal, com 199 verdadeiros positivos em várias classes, incluindo *high StNum*, *injection*, *inverse replay*, e *poisoned high rate*.

No entanto, há diferenças sutis entre os dois modelos. A classe *normal*, que no *Decision Tree* tinha 30 falsos negativos, apresenta um número maior de falsos negativos no *XGBoost*, com 33 instâncias incorretamente classificadas. Em contrapartida, a classe *random replay*, que tinha 4 falsos positivos no *Decision Tree*, apresenta menos erros no *XGBoost*, com apenas 2 falsos positivos. Portanto, apesar de ambos os modelos

demonstrarem alta eficácia na maioria das classes, o *Decision Tree* apresenta uma ligeira vantagem na classe *normal*, enquanto o *XGBoost* mostra uma melhoria na classe *random replay*. Então, não pode-se afirmar que o desempenho do *XGBoost* é notavelmente superior, mas sim que ambos os modelos possuem pontos fortes em diferentes áreas.

Por fim, na Figura 3c, é exibida a matriz de confusão para os resultados do classificador *Random Forest*. Percebe-se que sua performance foi perfeita para as classes rotuladas como *high StNum*, *injection*, e *poisoned High Rate* – assim como os demais classificadores. No entanto, para as amostras normais, gerou-se menos falsos positivos relacionados com a classe *masquerade fake fault* do que os demais classificadores.

Em seguida, na Seção 6.2 os atributos que apresentaram maior influência para os modelos de predições desses três algoritmos são apresentados.

6.2. Explicabilidade

A análise dos valores SHAP permite a geração de percepções que contribuem para o entendimento de quais informações estão sendo representativas para cada modelo de modo a orientar a sua tomada de decisão. A seguir, os modelos dos três algoritmos classificadores estudados são analisados e comparados por meio da ferramenta de explicabilidade *summary plot*. Os gráficos gerados por essa ferramenta utilizam valores SHAP para indicar a importância média de cada atributo no impacto da predição do modelo. Cada cor representa uma classe diferente, conforme a legenda. Note que os atributos omitidos da figura não possuem valores SHAP, portanto, não contribuem para os modelos dos algoritmos.

O gráfico de explicabilidade apresentado na Figura 4 ilustra os valores SHAP para o algoritmo *Decision Tree*. Para este algoritmo, o atributo `sqDiff` é o atributo com maior valor SHAP, sugerindo que a diferença entre o número de sequência (`sqNum`) de duas mensagens consecutivas é uma informação bastante relevante para a maioria dos ataques e em especial para os tipos *inverse replay* e *random replay*, onde apresenta alta magnitude. Logo em seguida, aparece o atributo `stNum`, que representa o número de status de um componente que está sendo monitorado. O alto valor SHAP dessas informações se devem ao fato de que as mensagens GOOSE que são transmitidas em subestações implementam um mecanismo baseado nesses atributos para sinalizar os dispositivos da rede sobre a normalidade (*i.e.,* situações estáveis) ou ocorrência de eventos (*e.g,* faltas elétricas). Em situações típicas, mensagens GOOSE são transmitidas em intervalos fixos, com o mesmo `stNum` e com um *sqNum* sendo incrementado a cada mensagem. Isto é, $sqDiff = 1$ e $stDiff = 0$. Em contraste, quando ocorrem eventos que precisam ser notificados na rede, as mensagens GOOSE usam o campo boleano `cbStatus` para sinalizar a mudança de estado de um dispositivo elétrico (*e.g.,* um disjuntor sendo fechado ou aberto). Nesse caso, o valor de `sqNum` para este evento é reiniciado e o valor de `stNum` é incrementado, portanto, são esperados valores $sqDiff < 0$ e $stDiff = 1$. Note que o campo $cbStatus$ que foi mencionado é o atributo que aparece em quarto lugar.

Destaca-se que o ataque `high_stNum` é facilmente detectado pela análise do atributo *StNum*, conforme esperado. Isso ocorre devido ao comportamento do atacante, que consiste em transmitir mensagens GOOSE com valores geralmente acima do esperado para este atributo, com o objetivo de causar uma negação de serviço. Consequentemente, este atributo aparece em segundo lugar em termos de importância geral. No entanto, observa-se que sua contribuição predominante é para a classe que representa o referido

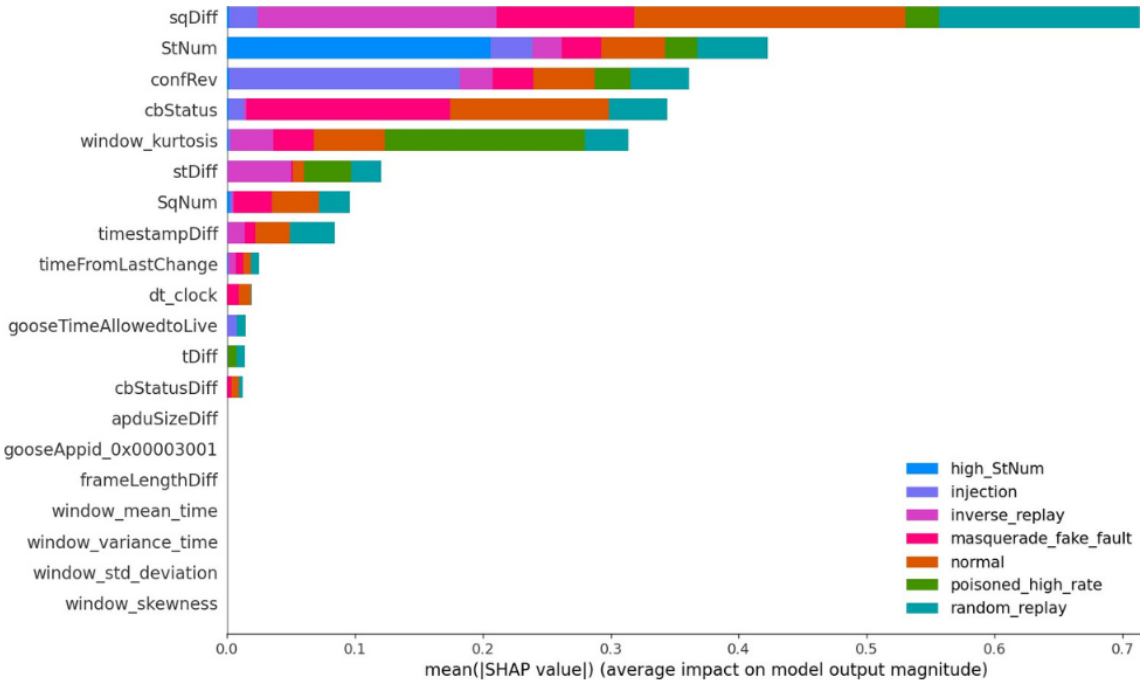


Figura 4. Valor SHAP global dos atributos no modelo gerado pelo *Decision Tree*.

ataque, cujo propósito é justamente a alteração desse atributo.

O atributo `confRev` não tem relação direta com o objetivo ou modo de operação do atacante em nenhuma das classes de ataques, no entanto, este é um atributo que carrega um valor específico da subestação vítima do ataque. Esse atributo é usado para sinalizar o receptor da mensagem que há alterações no campo *datSet*, que por sua vez carrega um conjunto de estados de carga útil incluindo o *cbStatus*. Portanto, essa informação permite gerar-se indícios de mensagens GOOSE carregando valores inconsistentes, que potencialmente, podem estar relacionadas com ataques. Isso é especialmente válido para o ataque *injection*, o qual apresenta um alto valor SHAP para este atributo.

Em quinto lugar, aparece o atributo `window_kurtosis`, o qual consiste em um atributo proposto originalmente neste trabalho. Isso demonstra que o método de enriquecimento de atributos empregado gerou informações relevantes, especialmente para o ataque *poisoned_high_rate*, no qual o atacante envia várias mensagens em um intervalo de tempo bastante curto. Para o modo de operação desse tipo de atacante, o atributo baseado em janelas de tempo se mostra totalmente adequado, inclusive, com valores SHAP superiores ao dobro dos valores dos demais atributos que já existiam no conjunto de dados analisados. Por fim, alguns outros atributos aparecem no gráfico com um valor SHAP inferior, mas ainda apresentam alguma contribuição, especialmente se tratando dos atributos relacionados com o carimbo de tempo da transmissão da mensagem (*i.e.,* `timestampDiff`, `timeFromLastChange`, `dtClock` e `tDiff`).

Tais observações demostram que o modelo *Decision Tree* se baseia fortemente em determinados atributos para diferenciar entre tipos de ataques e comportamento normal, sugerindo que a otimização e o foco nesses atributos podem melhorar ainda mais a precisão e o desempenho geral do modelo.

Para o classificador *XGBoost*, cujo gráfico de explicabilidade é apresentado na Figura 5, observa-se que o atributo `stNum` (que estava em segundo lugar para o *Decision Tree*) aparece como sendo o atributo mais impactante, especialmente na detecção do ataque *high_StNum*, refletindo sua relevância em identificar alterações críticas no status dos componentes. Ademais, em contraste com o algoritmo *Decision Tree*, para o *XGBoost* a diferença entre os atributos `timestamp` de duas mensagens consecutivas, isto é, `timestampDiff` aparece com um valor SHAP bastante alto. Destaca-se também o atributo `window_kurtosis`, que é particularmente relevante para detectar o ataque *poisoned_high_rate*, e o atributo `confRev`, que mostra uma forte contribuição na detecção de ataques de *injection*. Apesar dos valores SHAP variarem, alternando-se entre os atributos que aparecem com maior contribuição para a predição desse algoritmo em comparação com o analisado anteriormente, nota-se que, em geral, os subconjuntos dos atributos mais impactantes para ambos os algoritmos apresentam significativa interseção, exceto pelo atributo `dtClock`.

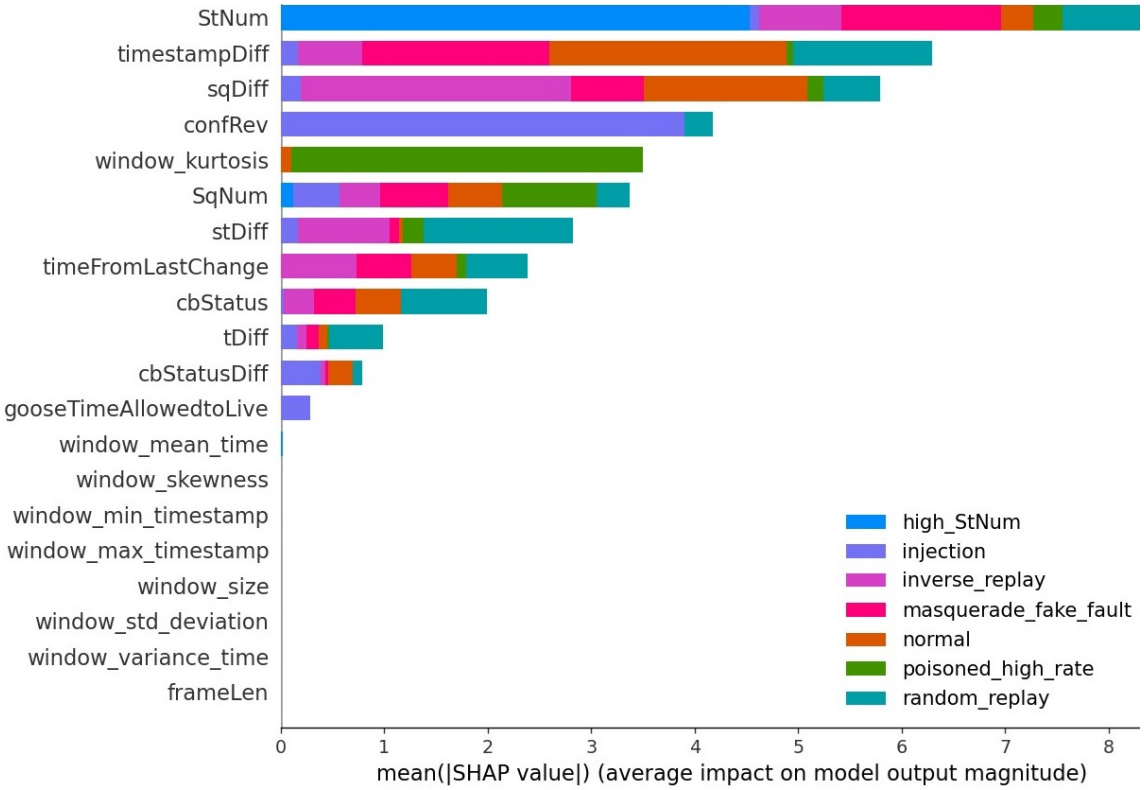


Figura 5. Importância global dos atributos no modelo gerado pelo *XGBoost*.

Ademais, comparando-se os gráficos das Figuras 4 e 5, observa-se que ambos os modelos atribuem alta importância aos atributos relacionadas à sequência de números e carimbos de tempo. No entanto, o *XGBoost* dá uma ênfase maior a `StNum` e `timestampDiff`, enquanto o *Decision Tree* atribui maior importância a `sqDiff`. Essas observações indicam que, embora ambos os modelos utilizem atributos semelhantes para diferenciar entre tipos de ataques e comportamento normal, eles podem estar focando em aspectos ligeiramente diferentes para melhorar a precisão das predições.

Interessantemente, o algoritmo *Random Forest*, além de demonstrar um impacto

alto para atributos semelhantes aos observados para os demais algoritmos, expande a lista de atributos com valores SHAP altos para outros atributos. Esses atributos adicionais incluem quatro atributos que foram propostos pelo processo de enriquecimento realizado neste trabalho, além do atributo `window_kurtosis`, que também foi proposto neste trabalho e usado pelos três algoritmos apresentados. Uma importante observação consiste no fato de que o *Random Forest* não apenas foi o único algoritmo a tirar proveito dessas informações, mas também foi o algoritmo mais preciso dentre os que foram estudados. Isso revela que, potencialmente, tais atributos tenham interferido para a geração de um menor número de falsos positivos.

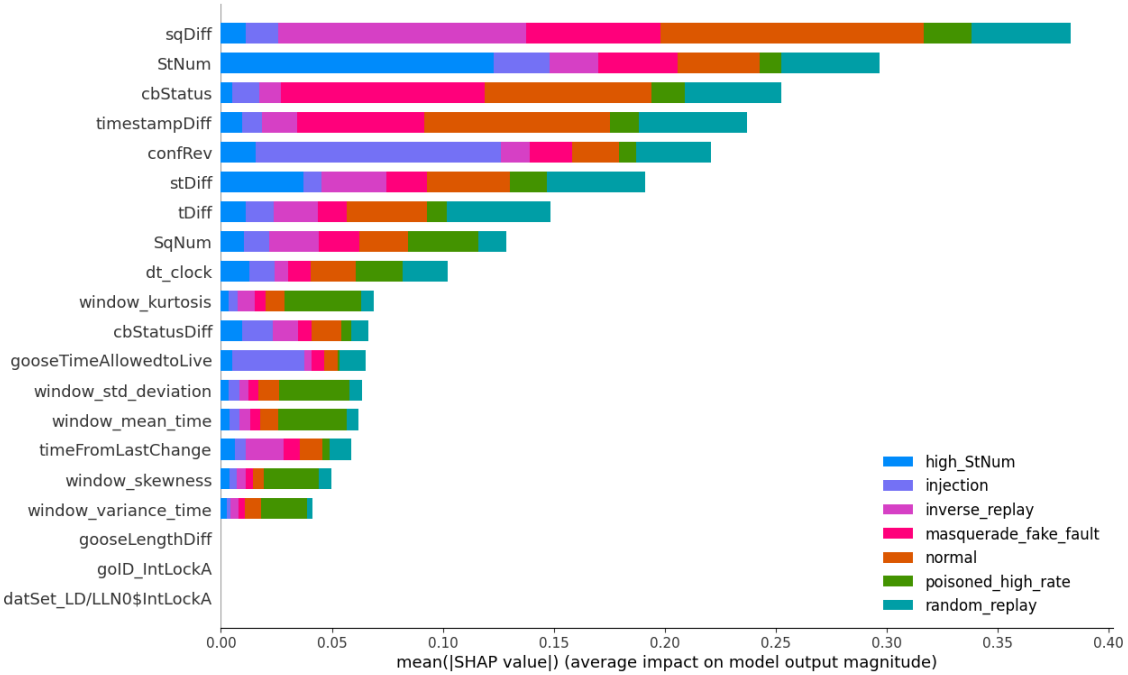


Figura 6. Importância global dos atributos para o modelo do *Random Forest*.

Essas diferenças no enfoque dos atributos sugerem que a combinação das abordagens de múltiplos modelos pode potencialmente levar a uma melhoria na detecção de ataques, otimizando a segurança e a performance do IDS. Ademais, em geral, o uso de XAI na detecção de intrusões traz implicações práticas significativas, permitindo que gestores de segurança compreendam melhor as decisões dos modelos de aprendizado de máquina. Essa transparência possibilita a identificação de atributos críticos para a detecção de ataques, ajudando a ajustar políticas de segurança, identificar vulnerabilidades, otimizar recursos e aprimorar continuamente as estratégias de proteção. Além disso, XAI tem potencial de facilitar a educação e treinamento da equipe de segurança e garante conformidade com normas regulamentares, ao proporcionar justificativas claras e auditáveis para as ações tomadas pelo sistema de detecção. No entanto, essa investigação foge do escopo do presente trabalho.

7. Considerações Finais

Neste trabalho, foi proposta uma arquitetura X-IDS para subestações elétricas, integrando técnicas de XAI e novos métodos de extração de atributos. O objetivo foi aumentar a

transparência e a confiabilidade dos IDSs tradicionais, permitindo uma melhor detecção e interpretação de ataques cibernéticos.

Os resultados experimentais demonstraram que o X-IDS proposto reduziu o viés em relação a certos ataques e aprimorou a interpretação de ataques complexos. As técnicas de XAI utilizadas forneceram explicações claras das decisões dos modelos de aprendizado de máquina, facilitando a análise de correções e novas implementações. Em particular, o classificador *Random Forest* apresentou as melhores métricas de desempenho, com acurácia e precisão de 98,79%, e revocação de 98,68%.

Como trabalhos futuros pretende-se explorar (i) métodos para otimizar a eficiência computacional das técnicas de XAI, (ii) a integração de dados em tempo real para melhorar a adaptabilidade e a precisão dos modelos, e (iii) novas técnicas de enriquecimento de atributos que possam contribuir para uma detecção robusta.

Referências

Bisong, E. and Bisong, E. (2019). Introduction to scikit-learn. *Building machine learning and deep learning models on google cloud platform: a comprehensive guide for beginners*, pages 215–229.

Davenport, T. H. (2018). *The AI advantage: How to put the artificial intelligence revolution to work*. mit Press.

Dresch, F. N., Scherer, F. H., Quincozes, S. E., and Kreutz, D. L. (2024). Modelos interpretáveis com inteligência artificial explicável (XAI) na detecção de intrusões em redes intra-veiculares controller area network (CAN). In *Anais do XIX Simpósio Brasileiro de Segurança da Informaçao e de Sistemas Computacionais*. SBC.

IEC, T. (2003). Communication networks and systems in substations. *IEC61850*.

Kuzlu, M., Cali, U., Sharma, V., and Guler, O. (2020). Gaining insight into solar photovoltaic power generation forecasting utilizing explainable artificial intelligence tools. *IEEE Access*, 8:187814–187823.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.

Molnar, C. (2022). *Interpretable Machine Learning*. Leanpub, 2 edition.

Munir, M. S., Shetty, S., and Rawat, D. B. (2023). Trustworthy artificial intelligence framework for proactive detection and risk explanation of cyber attacks in smart grid.

Neupane, S., Ables, J., Anderson, W., Mittal, S., Rahimi, S., Banicescu, I., and Seale, M. (2022). Explainable intrusion detection systems (x-ids): A survey of current methods, challenges, and opportunities. *IEEE Access*, 10:112392–112415.

Premaratne, U. K., Samarabandu, J., Sidhu, T. S., Beresh, R., and Tan, J.-C. (2010). An intrusion detection system for IEC61850 automated substations. *IEEE Transactions on Power Delivery*, 25(4):2376–2383.

Quincozes, S. E., Albuquerque, C., Passos, D., and Mossé, D. (2021). A survey on intrusion detection and prevention systems in digital substations. *Computer Networks*, 184:107679.

Quincozes, S. E., Albuquerque, C., Passos, D., and Mossé, D. (2023). ERENO: A Framework for Generating Realistic IEC–61850 Intrusion Detection Datasets for Smart Grids. *IEEE Transactions on Dependable and Secure Computing*.

Quincozes, V. E., Quincozes, S. E., Kazienko, J. F., Gama, S., Cheikhrouhou, O., and Koubaa, A. (2024). A survey on IoT application layer protocols, security challenges, and the role of explainable AI in IoT (XAIoT). *International Journal of Information Security*, 23(3):1975–2002.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). 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*, pages 1135–1144. ACM.

Sivamohan, S., Sridhar, S., and Krishnaveni, S. (2023). TEA-EKHO-IDS: An intrusion detection system for industrial CPS with trustworthy explainable AI and enhanced krill herd optimization. *Peer-to-Peer Networking and Applications*, 16(4):1993–2021.

Suaboot, J., Fahad, A., Tari, Z., Grundy, J., Mahmood, A. N., Almalawi, A., Zomaya, A. Y., and Drira, K. (2020). A taxonomy of supervised learning for IDSs in scada environments. *ACM Computing Surveys (CSUR)*, 53(2):1–37.

Vainio-Pekka, H., Agbese, M. O.-O., Jantunen, M., Vakkuri, V., Mikkonen, T., Rousi, R., and Abrahamsson, P. (2023). The role of explainable ai in the research field of ai ethics. *ACM Transactions on Interactive Intelligent Systems*, 13(4):1–39.

Wang, M., Zheng, K., Yang, Y., and Wang, X. (2020). An explainable machine learning framework for intrusion detection systems. *IEEE Access*, 8:73127–73141.

Youssef, T. A., El Hariri, M., Bugay, N., and Mohammed, O. (2016). Iec 61850: Technology standards and cyber-threats. In *2016 IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC)*, pages 1–6. IEEE.

Zolanvari, M., Yang, Z., Khan, K., Jain, R., and Meskin, N. (2021). TRUST XAI: Model-Agnostic Explanations for AI With a Case Study on IIoT Security. *IEEE internet of things journal*, 10(4):2967–2978.