Detector de Ataques Cibernéticos à Dispositivos IOT

Artur Correa[1], **Guilherme Martins**[1], **Lucas Azevedo**[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

arturcf00@gmail.com, gmsoares@inf.ufrgs.br, lucasdossantosa5@gmail.com

Abstract. *In an increasingly connected global landscape dependent on technological solutions for tasks of varying criticality, malicious actors seek to gain advantages through unauthorized access to data. In this context, cybersecurity has gained growing relevance. This work aims to build an intrusion detection model for IoT devices using a subset of the CIC IoT-DIAD 2024 dataset, which contains logs of 33 types of attacks applied to a topology of 105 IoT devices. The data underwent preprocessing to handle infinite and missing values. Two feature selection strategies were employed: one based on Pearson correlation (selecting protocol and TCP flag-related attributes) and another based on mutual information (selecting packet size and volume attributes). Supervised machine learning models—K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Logistic Regression, and Random Forest—were evaluated. Results indicate that KNN and Random Forest achieved the best performance in terms of accuracy, recall, and test AUC. However, KNN showed signs of overfitting, while Random Forest offered a better balance between detection and false positives. The choice between them may depend on specific computational efficiency requirements and the criticality of minimizing false negatives in a cybersecurity context.*

Resumo. *Num cenário global cada vez mais conectado e dependente de soluções tecnológicas para tarefas de diversos níveis de criticidade, agentes mal-intencionados buscam obter vantagens por meio de dados aos quais não têm direito. Nesse contexto, a cibersegurança tem ganhado relevância crescente. Este trabalho tem como objetivo construir um modelo de detecção de intrusões para dispositivos IoT utilizando um subconjunto do dataset CIC IoT-DIAD 2024, que reúne logs de 33 tipos de ataques aplicados a uma topologia com 105 dispositivos IoT. Os dados passaram por um pré-processamento para tratamento de valores infinitos e ausentes. Foram empregadas duas estratégias de seleção de atributos: uma baseada na correlação de Pearson (selecionando atributos relacionados a protocolo e flags TCP) e outra baseada em informação mútua (selecionando atributos de tamanho e volume de pacotes). Modelos supervisionados de aprendizado de máquina — K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Regressão Logística e Random Forest — foram avaliados. Os resultados indicam que KNN e Random Forest obtiveram os melhores desempenhos em termos de acurácia, recall e AUC de teste. Entretanto, o KNN apresentou indícios de overfitting, enquanto o Random Forest ofereceu um melhor equilíbrio entre detecção e falsos positivos. A escolha entre eles pode depender de requisitos específicos de eficiência computacional e da criticidade em minimizar falsos negativos num contexto de cibersegurança.*

1. Introdução

Em um cenário global cada vez mais conectado e dependente de soluções tecnológicas para tarefas de diversos níveis de criticidade, agentes mal-intencionados buscam obter vantagens por meio de dados aos quais não têm direito. Nesse contexto, a cibersegurança — definida como a prática de proteger computadores, redes, aplicações, dados e sistemas [Amazon Web Services 2023] — tem ganhado relevância crescente. Organizações de diferentes setores, que lidam com grandes volumes de informações confidenciais e precisam manter operações continuamente disponíveis, dependem de proteções robustas contra ameaças cibernéticas.

Segundo a [Brasscom 2025], o investimento projetado em cibersegurança no Brasil entre 2025 e 2028 será de R$ $104,06$ bilhões, enquanto o custo médio de uma violação de dados para empresas é de R$ $1,36$ milhão. As soluções de segurança atuam na proteção de dados, detecção de ameaças, autenticação de dispositivos de Internet das Coisas (IoT), criptografia de informações e identificação de ameaças avançadas. Tecnologias como big data, inteligência artificial, aprendizado de máquina, biometria comportamental, blockchain e computação quântica permitem prever ataques em tempo real, analisando indicadores e padrões de comportamento.

Os dispositivos de IoT (*Internet of Things*), em particular, têm experimentado adoção maciça. Com a redução no tamanho e no custo de sensores e processadores, tornou-se viável conectar à internet desde equipamentos industriais e médicos até itens do cotidiano, como cafeteiras, sistemas de iluminação residencial e refrigeradores [Alsamiri and Alsubhi 2019]. Esse fenômeno expande exponencialmente a superfície de ataque disponível a cibercriminosos. A rápida proliferação de dispositivos com capacidade computacional limitada — inferior até mesmo à de smartphones — exige o desenvolvimento de soluções de segurança baseadas em rede.

O objetivo deste trabalho é construir um modelo de detecção de intrusões com base em um subconjunto do dataset disponibilizado pelo departamento de cibersegurança da Universidade de New Brunswick, no Canadá. Esse conjunto reúne *logs* de 33 tipos de ataques aplicados a uma topologia com 105 dispositivos IoT. Em nosso experimento, utilizaremos uma versão reduzida desse dataset, o que permitirá analisar os impactos dessa redução no desempenho geral do modelo.

Um repositório no Github foi criado onde podem ser acessados tanto o notebook, quanto o dataset utilizado na criação deste artigo. Maiores instruções sobre o modo de usar também estarão neste repositório. Você poderá acessá-lo via este link: https://github.com/gui-ms/cmp263ML

2. Revisão Bibliográfica

Diversos trabalhos têm investigado o uso de aprendizado de máquina (AM) para detecção de ataques em ambientes de Internet das Coisas (IoT) e sistemas ciber-físicos (CPS). [Alsamiri and Alsubhi 2019] analisam o emprego de algoritmos de aprendizado de máquina na detecção de ataques em redes IoT, mostrando que classificadores supervisionados conseguem capturar padrões de tráfego malicioso a partir de atributos extraídos de comunicações entre dispositivos. Em uma linha semelhante, [Chora and Kozik 2014] aplicam técnicas de AM para identificação de ataques contra aplicações web, eviden-

ciando que abordagens baseadas em dados podem complementar ou superar soluções tradicionais baseadas em assinaturas.

[Fei and Shen 2023] apresentam uma revisão abrangente sobre o uso de aprendizado de máquina para proteger CPS contra ataques cibernéticos, discutindo tanto técnicas supervisionadas quanto não supervisionadas, além de abordagens profundas (*deep learning*). Os autores destacam desafios como a escassez de dados rotulados de alta qualidade, o desbalanceamento entre classes benignas e maliciosas e a dificuldade de lidar com cenários em constante evolução (*concept drift*). Estes desafios são ecoados por [Bouke et al. 2024], que revisam problemas de falta de dados (*data lack*), vazamento de informação (*data leakage*) e alta dimensionalidade em sistemas de detecção de intrusão, argumentando que muitos estudos da área não tratam adequadamente essas questões metodológicas, o que compromete a generalização dos modelos.

Além das soluções específicas de detecção, há uma base consolidada de literatura que discute aspectos fundamentais de pré-processamento, seleção de atributos e redução de dimensionalidade para aprendizado de máquina. [Fan et al. 2021] revisam técnicas de pré-processamento voltadas à descoberta de conhecimento em dados operacionais de edificações, enfatizando etapas como limpeza, transformação, normalização e tratamento de valores ausentes. [Nielsen 2021] propõe uma espécie de *checklist* de pré-processamento para construção de modelos de aprendizado de máquina, ressaltando que decisões tomadas nessa fase têm impacto direto no desempenho e na interpretabilidade dos classificadores. De forma complementar, a documentação do *scikit-learn* apresenta boas práticas e armadilhas comuns em projetos de AM [Scikit-learn a], alertando para riscos como vazamento de dados entre treino e teste, uso inadequado de validação cruzada e seleção de hiperparâmetros com viés. [Babu] discute, em linguagem acessível, estratégias de ajuste de hiperparâmetros, destacando o impacto da escolha correta de parâmetros sobre a capacidade de generalização dos modelos.

No que diz respeito à escolha e ao tratamento dos atributos, [Dhal and Azad 2021] apresentam uma pesquisa abrangente sobre seleção de atributos em diferentes domínios de aprendizado de máquina, destacando métodos *filter, wrapper e embedded*, bem como seus compromissos entre custo computacional e qualidade do subconjunto selecionado. Em paralelo, [Sorzano et al. 2014] e [Jia et al. 2022] revisam técnicas de redução de dimensionalidade, abordando métodos lineares (como Análise de Componentes Principais — PCA) e não lineares, além de suas aplicações em problemas de alto volume de atributos. A própria documentação do PCA no scikit-learn [Scikit-learn b] detalha o uso prático dessa técnica, incluindo aspectos de escalonamento, escolha do número de componentes e interpretação de variância explicada. Em cenários de detecção de intrusões, onde os conjuntos de dados frequentemente possuem centenas de atributos, essas técnicas são essenciais para reduzir redundâncias, mitigar o risco de *overfitting* e diminuir o custo de treinamento e inferência.

Outro ponto crucial para a construção de modelos confiáveis é a estratégia de validação. [Fontanari et al. 2022] discute estratégias de validação cruzada para bases balanceadas e desbalanceadas, mostrando que escolhas como *k-fold* simples, estratificado ou validações aninhadas (*nested cross-validation*) influenciam fortemente as estimativas de desempenho obtidas. Em problemas de segurança, nos quais eventos maliciosos tendem a ser raros em comparação ao tráfego legítimo, a adoção de estratégias de validação apropri-

adas e métricas além da acurácia (como *F1-score*, *recall* e curvas ROC/PR) é fundamental para evitar conclusões enviesadas.

Mais recentemente, observam-se esforços para disponibilizar conjuntos de dados públicos que permitam avaliar, de forma comparável, diferentes abordagens de detecção de anomalias em ambientes IoT. O conjunto CIC IoT-DIAD 2024 foi proposto pelo *Canadian Institute for Cybersecurity* como um *dataset* de dupla função, permitindo tanto a identificação de dispositivos IoT quanto a detecção de anomalias em tráfego de rede [for Cybersecurity 2024]. [Rabbani et al. 2024] exploram esse conjunto de dados para propor técnicas de identificação de dispositivos e detecção de anomalias em ambientes IoT, destacando cenários realistas com múltiplos tipos de dispositivos e diferentes perfis de tráfego. Trabalhos dessa natureza contribuem para aproximar a pesquisa acadêmica de situações práticas, ao mesmo tempo em que estabelecem benchmarks que facilitam a comparação entre modelos e *pipelines* de aprendizado de máquina.

No plano conceitual, definições gerais de segurança cibernética fornecidas por provedores de nuvem, como a [Amazon Web Services 2023], reforçam a ideia de que a proteção de ativos digitais envolve camadas complementares de defesa, incluindo controles preventivos, detectivos e corretivos. Quando combinadas com diagnósticos de associações do setor de TIC (Tecnologia da Informação e Comunicação), como a [Brasscom 2025], essas visões ajudam a contextualizar o papel da detecção automatizada de ataques dentro de uma estratégia mais ampla de gestão de riscos cibernéticos, especialmente em setores com forte dependência de serviços digitais.

Em síntese, a literatura aponta para um crescente amadurecimento das técnicas de aprendizado de máquina aplicadas à detecção de ataques em IoT, CPS e aplicações web [Alsamiri and Alsubhi 2019, Chora and Kozik 2014, Almajed et al. 2022, Fei and Shen 2023, Miao et al. 2021]. Ao mesmo tempo, revisões recentes destacam lacunas metodológicas importantes, sobretudo no tratamento de problemas de alta dimensionalidade, desbalanceamento e vazamento de dados em sistemas de detecção de intrusão [Bouke et al. 2024, Dhal and Azad 2021, Sorzano et al. 2014, Jia et al. 2022, Fontanari et al. 2022, Scikit-learn a]. Há, portanto, espaço para estudos que integrem de forma sistemática: (i) boas práticas de pré-processamento de dados [Fan et al. 2021, Nielsen 2021], (ii) técnicas de seleção e redução de atributos [Dhal and Azad 2021, Sorzano et al. 2014, Jia et al. 2022, Scikit-learn b], (iii) estratégias robustas de validação e ajuste de hiperparâmetros [Fontanari et al. 2022, Babu , Scikit-learn a] e (iv) uso de bases recentes e desafiadoras, como o CIC IoT-DIAD 2024 [for Cybersecurity 2024, Rabbani et al. 2024]. O trabalho desenvolvido neste estudo se insere nesse cenário, buscando dialogar com essas recomendações e contribuir com uma avaliação mais rigorosa e reprodutível de modelos de detecção de anomalias em ambientes conectados.

3. Desenvolvimento

Nesta seção são descritas as etapas de desenvolvimento do detector de ataques a dispositivos IoT baseado em aprendizado de máquina. Apresentam-se o conjunto de dados utilizado e o processo de preparação dos *flows*, as transformações aplicadas aos atributos, a definição dos modelos supervisionados avaliados e a estratégia de validação adotada. O objetivo é explicitar, de forma reprodutível, todas as decisões metodológicas que compõem o pipeline de detecção de tráfego malicioso.

3.1. Conjunto de dados CIC IoT-DIAD 2024

Para avaliar o detector de ataques a dispositivos IoT, foi utilizado o *CIC IoT-DIAD 2024*, um conjunto de dados público mantido pelo Canadian Institute for Cybersecurity (*CIC*) e pela University of New Brunswick [for Cybersecurity 2024, Rabbani et al. 2024]. O dataset é descrito como *dual*, pois combina atributos voltados tanto à identificação de dispositivos IoT quanto à caracterização de fluxos de rede (*flows*) para detecção de anomalias. Neste trabalho, utiliza-se exclusivamente a porção de *flows*, que está diretamente alinhada com o objetivo de detecção de tráfego malicioso.

A partir dos arquivos disponibilizados pelo CIC, foi construída uma amostra contendo apenas fluxos rotulados como benignos ou maliciosos. O subconjunto final utilizado neste estudo possui 220 406 registros de fluxo e 84 atributos por fluxo, incluindo taxas, contadores e medidas temporais. Esses atributos são majoritariamente numéricos, acompanhados de uma coluna categórica responsável pelo rótulo de classe, e resultam em um arquivo de aproximadamente 114 MB em disco.

A coluna `Label` indica se cada fluxo é benigno (`0`) ou malicioso (`1`). Na amostra adotada, a distribuição é moderadamente desbalanceada, com cerca de 118 mil fluxos maliciosos e 91 mil fluxos benignos. Esse desbalanceamento reforça a importância de métricas sensíveis à classe minoritária na etapa de avaliação, como *recall* e F1-score da classe de ataque, e orienta as escolhas feitas nas etapas posteriores do pipeline.

A montagem do dataset usado neste trabalho se deu da seguinte maneira:

- 91280 Entradas de tráfego benigno;
- 9999 Entradas de tráfego DDOS-HTTP;
- 45421 Entradas de tráfego DDOS-ICMP;
- 3620 Entradas de tráfego Dictionary Brute Force;
- 9999 Entradas de tráfego DNS-Spoofing;
- 9999 Entradas de tráfego DOS-HTTP;
- 9999 Entradas de tráfego DOS-UDP;
- 20099 Entradas de tráfego Mirai;
- 9999 Entradas de tráfego MITM-ARP-Spoofing;
- 9999 Entradas de tráfego Vulnerability Scan;

Os arquivos que possuíam muitas linhas, com exceção do DDOS-ICMP e Mirai, foram truncados usando o comando nativo do linux `sed -i '10000,$d' arquivo.csv`. Outra alteração necessária foi alterar o valor que estava na coluna *label*, visto que ao baixar o arquivo .csv o que se encontra nessa coluna é "*NeedManualLabel*", portanto utilizamos outro comando do linux, o `sed -i 's/valor_antigo/valor_novo/g' arquivo.csv` substituindo *NeedManualLabel* por 0 ou 1.

No total, o CIC IoT-DIAD 2024 contempla 33 ataques distintos executados sobre a topologia IoT, organizados em sete grandes tipos de ameaças: ataques de negação distribuída de serviço (DDoS), negação de serviço (DoS), atividades de reconhecimento (*Reconnaissance*, que incluem varreduras e coleta de informações), ataques baseados na Web, ataques de força bruta, ataques de *spoofing* e ataques de botnet Mirai. Esses cenários geram padrões de tráfego bastante heterogêneos, o que torna o conjunto particularmente adequado para avaliar detectores de anomalias em ambientes IoT com múltiplos vetores de ataque.

3.2. Tratamento e pré-processamento dos dados

O pré-processamento teve como foco garantir que todas as variáveis pudessem ser consumidas por algoritmos de aprendizado de máquina supervisionado sem a presença de valores indefinidos. A análise inicial mostrou que os principais problemas se concentravam em duas variáveis derivadas pelo *CICFlowMeter*: `Flow Bytes/s` e `Flow Packets/s`, responsáveis por representar, respectivamente, as taxas de bytes e de pacotes por segundo em cada fluxo.

Foram identificados 2 986 registros com pelo menos um valor infinito (`inf`) em `Flow Bytes/s` ou `Flow Packets/s`, além de 2 898 ocorrências de valores ausentes (`NaN`) em `Flow Bytes/s`. Mais de 98% desses casos estavam associados à classe `Label = 1`, isto é, fluxos rotulados como maliciosos. Em geral, esses problemas estão relacionados a fluxos com duração extremamente curta ou contagens muito pequenas de pacotes, o que leva a divisões por zero ou por valores próximos de zero no cálculo das taxas, resultando em valores infinitos ou indefinidos.

3.3. Estratégias de tratamento

Para preservar o maior número possível de instâncias e, ao mesmo tempo, manter valores numericamente plausíveis, optou-se por substituir os valores problemáticos por máximos finitos observados no próprio conjunto de dados. Os valores infinitos em `Flow Bytes/s` foram substituídos pelo maior valor finito observado nessa coluna, aproximadamente $2{,}9 \times 10^9$ bytes por segundo. Os valores infinitos em `Flow Packets/s` foram substituídos pelo maior valor finito observado na coluna, aproximadamente 3×10^6 pacotes por segundo. As ocorrências de `NaN` em `Flow Bytes/s` foram tratadas da mesma forma, recebendo o mesmo valor máximo utilizado para os casos de infinito.

Essa decisão segue três princípios. Primeiro, a preservação das instâncias: evita-se a remoção de quase três mil fluxos, muitos deles maliciosos, o que seria especialmente prejudicial em um cenário já desbalanceado. Segundo, a coerência prática: fluxos associados a ataques tendem, de fato, a apresentar taxas muito elevadas de bytes ou pacotes por segundo, de modo que aproximá-los pelos maiores valores observados é razoável do ponto de vista operacional. Por fim, a coerência estatística: ao limitar os valores a máximos já presentes nos dados, evita-se a introdução de outliers artificiais e mantém-se a forma geral da distribuição empírica das variáveis.

Após a aplicação dessa estratégia de tratamento, todas as instâncias do subconjunto foram preservadas e as colunas `Flow Bytes/s` e `Flow Packets/s` passaram a conter apenas valores numéricos finitos, sem ocorrência de `inf` ou `NaN`. O conjunto resultante encontra-se, assim, em condição adequada para as etapas seguintes do pipeline, que incluem normalização ou reescala das variáveis, eventual redução de dimensionalidade e treinamento dos modelos de aprendizado de máquina focados na detecção de tráfego malicioso em ambientes IoT.

3.4. Redução de dimensionalidade e seleção de atributos

Após o tratamento das inconsistências, o conjunto de dados permaneceu com 84 atributos numéricos associados a cada *flow*. Embora seja possível treinar modelos diretamente com todas essas variáveis, a presença de atributos redundantes ou pouco informativos pode aumentar o custo computacional e favorecer *overfitting*. Por isso, foram testadas

duas estratégias complementares de redução de dimensionalidade baseadas em filtros: correlação de Pearson e seleção por informação mútua.

Na primeira estratégia, consideraram-se apenas as colunas numéricas e calculou-se a correlação de Pearson entre cada atributo e o rótulo `Label`. Em seguida, foram selecionados os seis atributos com maior correlação em valor absoluto com a classe, resultando em um subconjunto composto por variáveis associadas ao protocolo, à duração do fluxo e à contagem de *flags* TCP (*Transmission Control Protocol*), como `Protocol`, `Flow_Duration`, `SYN_Flag_Count`, `RST_Flag_Count`, `Fwd_IAT_Total` e `FWD_Init_Win_Bytes`. Nesse subconjunto, foram avaliadas todas as combinações possíveis de uma a seis variáveis por meio de um modelo de regressão logística generalizada (GLM binomial). Para cada combinação, o conjunto foi dividido de forma estratificada em treino e validação, reservando-se um subconjunto fixo de 27 000 *flows* para validação. As variáveis de entrada foram normalizadas com *Min-Max scaling*, o conjunto de treino foi balanceado com a *Synthetic Minority Over-sampling Technique* (*SMOTE*), e o modelo GLM foi ajustado usando função de ligação logística. A métrica de comparação foi a área sob a curva ROC (AUC) no conjunto de validação. As melhores combinações incluíam consistentemente as variáveis `RST_Flag_Count`, `Protocol`, `SYN_Flag_Count` e `Flow_Duration`, que foram selecionadas como primeiro grupo de atributos para os experimentos com modelos de aprendizado de máquina:

	Variáveis	AUC Treino	AUC Teste
2	('Fwd_IAT_Total',)	0.537337	0.543932
16	('Fwd_IAT_Total', 'Flow_Duration')	0.540667	0.546455
4	('Flow_Duration',)	0.569291	0.577003
5	('FWD_Init_Win_Bytes',)	0.582379	0.577412
19	('SYN_Flag_Count', 'FWD_Init_Win_Bytes')	0.593165	0.589632
3	('SYN_Flag_Count',)	0.591562	0.592476

Table 1: 6 Valores mais baixos

	Variáveis	AUC Treino	AUC Teste
41	('RST_Flag_Count', 'Protocol', 'Fwd_IAT_Total', 'SYN_Flag_Count')	0.691747	0.696046
28	('RST_Flag_Count', 'SYN_Flag_Count', 'Flow_Duration')	0.692448	0.697301
50	('RST_Flag_Count', 'SYN_Flag_Count', 'Flow_Duration', 'FWD_Init_Win_Bytes')	0.693767	0.697331
46	('RST_Flag_Count', 'Protocol', 'Flow_Duration', 'FWD_Init_Win_Bytes')	0.700516	0.702937
59	('RST_Flag_Count', 'Protocol', 'SYN_Flag_Count', 'Flow_Duration', 'FWD_Init_Win_Bytes')	0.701434	0.704564
44	('RST_Flag_Count', 'Protocol', 'SYN_Flag_Count', 'Flow_Duration')	0.702184	0.705656

Table 2: 6 Valores mais altos

Na segunda estratégia, buscou-se identificar atributos informativos a partir da perspectiva de teoria da informação. Para isso, foi utilizada a função `SelectKBest` com o critério *mutual information* (`mutual_info_classif`) aplicada ao conjunto completo de variáveis numéricas. Foram selecionados seis atributos com maior informação mútua em relação ao rótulo, resultando em um subconjunto dominado por medidas de tamanho de pacotes e volume de dados, como `Total_Length_of_Fwd_Packet`, `Total_Length_of_Bwd_Packet`, `Packet_Length_Max`, `Packet_Length_Mean` e `Average_Packet_Size`. De forma análoga ao procedimento anterior, avaliou-se o desempenho de um GLM para combinações dessas variáveis, novamente com normalização, balanceamento via *SMOTE* e AUC em um subconjunto de validação. Embora os valores de AUC obtidos com esse grupo tenham sido ligeiramente inferiores aos do grupo baseado em *flags* TCP, o comportamento das variáveis de tamanho de pacote mostrou-se complementar. Assim, o melhor subconjunto de atributos de comprimento de pacotes foi mantido como segundo grupo de variáveis para a etapa de comparação entre modelos de aprendizado de máquina.

	Variáveis	AUC Treino	AUC Teste
38	('Total_Length_of_Bwd_Packet', 'Packet_Length_Max', 'Average_Packet_Size')	0.422683	0.422048
19	('Packet_Length_Max', 'Average_Packet_Size')	0.422731	0.422071
52	('Total_Length_of_Fwd_Packet', 'Total_Length_of_Bwd_Packet', 'Packet_Length_Max', 'Average_Packet_Size')	0.424136	0.423277
35	('Total_Length_of_Fwd_Packet', 'Packet_Length_Max', 'Average_Packet_Size')	0.424166	0.423340
37	('Total_Length_of_Bwd_Packet', 'Packet_Length_Max', 'Packet_Length_Mean')	0.425561	0.425200

Continua na próxima página

	Variáveis	AUC Treino	AUC Teste
18	('Packet_Length_Max', 'Packet_Length_Mean')	0.425558	0.425216

Table 3: 6 Valores mais baixos

	Variáveis	AUC Treino	AUC Teste
15	('Total_Length_of_Bwd_Packet', 'Packet_Length_Max')	0.541175	0.545520
3	('Packet_Length_Max',)	0.542061	0.546537
31	('Total_Length_of_Fwd_Packet', 'Total_Length_of_Bwd_Packet', 'Packet_Length_Max')	0.543143	0.547861
12	('Total_Length_of_Fwd_Packet', 'Packet_Length_Max')	0.543095	0.547879
11	('Total_Length_of_Fwd_Packet', 'Total_Length_of_Bwd_Packet')	0.568550	0.573607
1	('Total_Length_of_Fwd_Packet',)	0.569280	0.574628

Table 4: 6 Valores mais altos

Dessa forma, os experimentos subsequentes foram conduzidos com dois cenários de entrada: um cenário baseado em atributos de protocolo e *flags* (`RST_Flag_Count`, `Protocol`, `SYN_Flag_Count`, `Flow_Duration`) e um cenário baseado em medidas de tamanho e volume de pacotes (`Total_Length_of_Fwd_Packet`, `Total_Length_of_Bwd_Packet`).

3.5. Modelos de aprendizado de máquina e estratégia de validação

Com os grupos de atributos definidos, foram avaliados quatro classificadores supervisionados amplamente utilizados em detecção de intrusões: *k*-Nearest Neighbours (KNN), Support Vector Machine (SVM) linear, Regressão Logística e Random Forest. Todos os modelos foram implementados em Python utilizando as bibliotecas `scikit-learn`, `imbalanced-learn` e `statsmodels`, em ambiente Google Colab.

Em todos os experimentos de comparação entre modelos, o conjunto de dados pré-processado foi dividido de forma estratificada em treino e teste, reservando-se 5 000 *flows* para compor o conjunto de teste e utilizando o restante para treinamento. Antes do ajuste dos classificadores, as variáveis de entrada foram normalizadas com *Min-Max scaling*, de modo a restringir todas as *features* ao intervalo $[0, 1]$. Na sequência, o conjunto de treino foi balanceado com a técnica SMOTE, que gera instâncias sintéticas da classe minoritária a partir de vizinhos mais próximos, resultando em um conjunto de treino aproximadamente balanceado entre fluxos benignos e maliciosos. O conjunto de teste permaneceu inalterado, refletindo o desbalanceamento original do problema.

Os modelos foram configurados com hiperparâmetros escolhidos para equilibrar desempenho e custo computacional. O classificador KNN utilizou pesos proporcionais à distância (*distance weighting*) e número de vizinhos definido de forma heurística a

partir do tamanho do conjunto de treino, limitado a um intervalo reduzido para evitar valores excessivamente grandes. A SVM foi implementada como um classificador linear (`LinearSVC`) com $C = 0{,}5$ e *max_iter* elevado, envolvido por um *wrapper* de calibração de probabilidades (`CalibratedClassifierCV`) para permitir o cálculo de métricas baseadas em escores contínuos, como a AUC. A Random Forest foi configurada com 100 árvores, profundidade máxima limitada, restrições mínimas de divisão e número mínimo de amostras por folha para conter o crescimento das árvores e com *class_weight* ajustado para *balanced*, de forma a penalizar mais erros na classe minoritária. A Regressão Logística foi ajustada com o otimizador `lbfgs` e limite de 1 000 iterações.

Para cada combinação de grupo de atributos e modelo de aprendizado de máquina, foram calculadas métricas no conjunto de treino e no conjunto de teste. As principais métricas consideradas foram a área sob a curva ROC (AUC), a acurácia global e o *recall* da classe de ataque, além da matriz de confusão, a partir da qual se obtêm os totais de verdadeiros positivos, verdadeiros negativos, falsos positivos e falsos negativos. Essas métricas são utilizadas na Seção 4 para comparar criticamente o comportamento dos modelos, tanto em termos de capacidade de detectar ataques quanto em termos de equilíbrio entre detecção e geração de falsos alarmes.

4. Discussão de Resultados e Conclusão

Foram obtidas métricas para as duas estratégias de seleção de características. Em uma das tabelas são apresentados os valores das principais métricas — acurácia (acc), AUC de treino (tn) e teste (ts), e recall —, enquanto a outra exibe os valores da matriz de confusão. Vale notar que, de acordo com a estrutura deste *dataset*, um Verdadeiro Positivo (TP) corresponde à identificação correta de tráfego malicioso, e um Verdadeiro Negativo (TN) à classificação correta de tráfego benigno.

4.1. Avaliação Estratégia Correlação de Pearson:

	Modelo	Variáveis	AUC Tn	AUC Ts	Recall Ts	Acc Ts
0	LogReg	['RST_Flag', 'Protocol', 'SYN_Flag', 'Flow_Dur']	0.702	0.711	0.442	0.619
1	RandomForest	['RST_Flag', 'Protocol', 'SYN_Flag', 'Flow_Dur']	0.833	0.833	0.637	0.732
2	KNN	['RST_Flag', 'Protocol', 'SYN_Flag', 'Flow_Dur']	0.998	0.782	0.722	0.719
3	SVM	['RST_Flag', 'Protocol', 'SYN_Flag', 'Flow_Dur']	0.706	0.715	0.482	0.625

Table 5. Métricas de Performance dos Modelos

	Modelo	Variáveis	TP	TN	FP	FN
0	LogReg	['RST_Flag', 'Protocol', 'SYN_Flag', 'Flow_Dur']	1296	1797	274	1633
1	RandomForest	['RST_Flag', 'Protocol', 'SYN_Flag', 'Flow_Dur']	1865	1796	275	1064
2	KNN	['RST_Flag', 'Protocol', 'SYN_Flag', 'Flow_Dur']	2116	1478	593	813
3	SVM	['RST_Flag', 'Protocol', 'SYN_Flag', 'Flow_Dur']	1413	1710	361	1516

Table 6. Matriz de Confusão dos Modelos

4.2. Avaliação Estratégia Informação Mútua:

	Modelo	Variáveis	AUC Tn	AUC Ts	Recall Ts	Acc Ts
0	LogReg	['Fwd_Pkt_Len', 'Bwd_Pkt_Len']	0.500	0.500	0.000	0.414
1	RandomForest	['Fwd_Pkt_Len', 'Bwd_Pkt_Len']	0.838	0.843	0.680	0.754
2	KNN	['Fwd_Pkt_Len', 'Bwd_Pkt_Len']	0.910	0.890	0.855	0.837
3	SVM	['Fwd_Pkt_Len', 'Bwd_Pkt_Len']	0.570	0.564	0.954	0.567

Table 7. Métricas de Performance dos Modelos

	Modelo	Variáveis	TP	TN	FP	FN
0	LogReg	['Fwd_Pkt_Len', 'Bwd_Pkt_Len']	0	2071	0	2929
1	RandomForest	['Fwd_Pkt_Len', 'Bwd_Pkt_Len']	1993	1775	296	936
2	KNN	['Fwd_Pkt_Len', 'Bwd_Pkt_Len']	2504	1679	392	425
3	SVM	['Fwd_Pkt_Len', 'Bwd_Pkt_Len']	2795	42	2029	134

Table 8. Matriz de Confusão dos Modelos

4.3. Conclusão

Para as variáveis selecionadas pela correlação de Pearson, observa-se uma vantagem dos modelos KNN e *Random Forest* em relação aos demais em termos de acurácia, *recall* e AUC de teste. Entretanto, o KNN apresenta indícios de *overfitting*, dada a disparidade entre as AUCs de treino (0.998) e teste (0.782). A análise da matriz de confusão, aliada à AUC de teste, sugere uma ligeira vantagem do *Random Forest* na razão entre Verdadeiros Positivos e Falsos Positivos — ou seja, menor tendência a classificar tráfego benigno como malicioso. No entanto, em cenários de cibersegurança, é mais crítico evitar que tráfego malicioso seja ignorado (Falsos Negativos). Nesse aspecto, o KNN destaca-se em relação aos demais.

Quanto às variáveis selecionadas por informação mútua, a Regressão Logística apresentou desempenho equivalente a um chute aleatório (AUC $= 0.500$). Por outro lado, KNN e *Random Forest* mantiveram bons resultados, com essa seleção de atributos possibilitando ainda a redução de Falsos Negativos e o aumento da acurácia em ambos os modelos.

Diante do exposto, conclui-se que, considerando apenas as métricas, o KNN é o modelo de melhor desempenho. Ressalta-se, porém, que tanto o KNN quanto o SVM demandaram maior tempo de classificação. Dessa forma, a depender dos requisitos de eficiência computacional, o *Random Forest* pode ser uma alternativa mais adequada.

5. Próximos Passos

Próximos passos incluem a classificação do tipo de ataque, e uso de gpus para aceleração de treino. Para a classificação, pode-se adicionar uma nova coluna alvo que informa o tipo de ataque, ou aplicar algum método de aprendizado não-supervisionado, enquanto a biblioteca cuda do python oferece funções que permitem o uso de gpus para acelerar o tempo de execução de códigos.

References

[Almajed et al. 2022] Almajed, R., Ibrahim, A., Abualkishik, A. Z., et al. (2022). Using machine learning algorithm for detection of cyber-attacks in cyber physical systems. *Periodicals of Engineering and Natural Sciences (PEN)*, 10(3):261–275.

[Alsamiri and Alsubhi 2019] Alsamiri, J. and Alsubhi, K. (2019). Internet of things cyber attacks detection using machine learning. *International Journal of Advanced Computer Science and Applications*, 10(12).

[Amazon Web Services 2023] Amazon Web Services (2023). O que é segurança cibernética? – explicação sobre segurança cibernética – AWS. Amazon Web Services. Disponível em: https://aws.amazon.com/pt/what-is/cybersecurity/.

[Babu] Babu, A. A comprehensive guide to hyperparameter tuning in machine learning. Medium. Disponível em: https://medium.com/@aditib259/a-comprehensive-guide-to-hyperparameter-tuning-in-machine-learning-dc

[Bouke et al. 2024] Bouke, M., Abdullah, A., Udzir, N., et al. (2024). Overcoming the challenges of data lack, leakage, and dimensionality in intrusion detection systems: A comprehensive review. *Journal of Communication and Information Systems*, 39(2024):22–34.

[Brasscom 2025] Brasscom (2025). Relatório de cibersegurança 2025: Panorama e insights - brasscom. Disponível em: https://brasscom.org.br/pdfs/relatorio-de-ciberseguranca-2025-panorama-e-insights/. Acessado em: [Data de acesso].

[Chora and Kozik 2014] Chora, M. and Kozik, R. (2014). Machine learning techniques applied to detect cyber attacks on web applications. *Logic Journal of IGPL*, 23(1):45–56.

[Dhal and Azad 2021] Dhal, P. and Azad, C. (2021). A comprehensive survey on feature selection in the various fields of machine learning. *Applied Intelligence*, 52.

[Fan et al. 2021] Fan, C., Chen, M., Wang, X., et al. (2021). A review on data preprocessing techniques toward efficient and reliable knowledge discovery from building operational data. *Frontiers in Energy Research*, 9.

[Fei and Shen 2023] Fei, C. and Shen, J. (2023). Machine learning for securing cyber-physical systems under cyber attacks: A survey. *Franklin Open*, 4:100041–100041.

[Fontanari et al. 2022] Fontanari, T., Fróes, T. C., and Recamonde-Mendoza, M. (2022). Cross-validation strategies for balanced and imbalanced datasets. In *Lecture Notes in Computer Science*, pages 626–640.

[for Cybersecurity 2024] for Cybersecurity, C. I. (2024). CIC IoT-DIAD 2024 dataset: A dual-function dataset for IoT device identification and anomaly detection. https://www.unb.ca/cic/datasets/iot-diad-2024.html. Acesso em: 30 set. 2025.

[Jia et al. 2022] Jia, W., Sun, M., Lian, J., et al. (2022). Feature dimensionality reduction: a review. *Complex & Intelligent Systems*, 8:2663–2693.

[Miao et al. 2021] Miao, Y., Chen, C., Pan, L., et al. (2021). Machine learning-based cyber attacks targeting on controlled information. *ACM Computing Surveys*, 54(7):1–36.

[Nielsen 2021] Nielsen, L. (2021). *A Checklist for Data pre-processing before you build your Machine Learning Model A list of things-to-do for data pre-processing for creating Machine Learning data-sets (and a few handy TIPS).* Towards Data Science.

[Rabbani et al. 2024] Rabbani, M., Gui, J., Nejati, F., Zhou, Z., Kaniyamattam, A., Mirani, M., Piya, G., Opushnyev, I., Lu, R., and Ghorbani, A. A. (2024). Device identification and anomaly detection in iot environments. *IEEE Internet of Things Journal*. Artigo associado ao dataset CIC IoT-DIAD 2024.

[Scikit-learn a] Scikit-learn. *Common pitfalls and recommended practices*. Disponível em: `https://scikit-learn.org/stable/common_pitfalls.html`.

[Scikit-learn b] Scikit-learn. *sklearn.decomposition.PCA — scikit-learn 0.20.3 Documentation*. Disponível em: `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html`.

[Sorzano et al. 2014] Sorzano, C., Vargas, J., and Pascual-Montano, A. (2014). A survey of dimensionality reduction techniques.