UNIVERSIDADE FEDERAL DO CEARÁ

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

MESTRADO ACADÊMICO EM ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

GERALDO EUFRAZIO MARTINS JÚNIOR

POSEIDDON: PLATAFORMA DE SUPERVISÃO E DETECÇÃO DE INTRUSÕES DOS/DDOS ORIENTADA A NÓS DE BORDA.

SOBRAL

2025

GERALDO EUFRAZIO MARTINS JÚNIOR

POSEIDDON: PLATAFORMA DE SUPERVISÃO E DETECÇÃO DE INTRUSÕES DOS/DDOS ORIENTADA A NÓS DE BORDA.

Dissertação apresentada ao Curso de Mestrado Acadêmico em Engenharia Elétrica e de Computação do Programa de Pós-Graduação em Engenharia Elétrica e de Computação da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia Elétrica e Computação. Área de Concentração: Engenharia da Computação

Orientador: Prof. Dr. Wendley Souza da Silva

SOBRAL

2025

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

M343p Martins Júnior, Geraldo Eufrazio.
POSEIDDON: PLATAFORMA DE SUPERVISÃO E DETECÇÃO DE INTRUSÕES DOS/DDOS ORIENTADA A NÓS DE BORDA. / Geraldo Eufrazio Martins Júnior. – 2025.
70 f. : il. color.

Dissertação (mestrado) – Universidade Federal do Ceará, Campus de Sobral, Programa de Pós-Graduação em Engenharia Elétrica e de Computação, Sobral, 2025.
Orientação: Prof. Dr. Wendley Souza da Silva.

1. Edge computing. 2. Machine Learning. 3. DDoS. 4. DoS. 5. IoT. I. Título.

CDD 621.3

GERALDO EUFRAZIO MARTINS JÚNIOR

POSEIDDON: PLATAFORMA DE SUPERVISÃO E DETECÇÃO DE INTRUSÕES DOS/DDOS ORIENTADA A NÓS DE BORDA.

Dissertação apresentada ao Curso de Mestrado Acadêmico em Engenharia Elétrica e de Computação do Programa de Pós-Graduação em Engenharia Elétrica e de Computação da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia Elétrica e Computação. Área de Concentração: Engenharia da Computação

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. Wendley Souza da Silva (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Israel Eduardo Barros Filho
Universidade Federal do Ceará (UFC)

Prof. Dr. Antonio Emerson Barros Tomaz
Universidade Federal do Ceará (UFC)

A toda a minha família, pelo apoio incondicional.

AGRADECIMENTOS

Inicio esta página com o coração grato, primeiramente a Deus, por ter sido a força motriz que guiou e iluminou cada passo desta jornada. Pelo dom da vida e pela resiliência concedida para superar os desafios, visíveis e invisíveis, que se apresentaram ao longo deste caminho. A Ele, minha eterna gratidão.

A minha trajetória acadêmica foi pavimentada por instituições de excelência. Ao Instituto Federal do Ceará (IFCE), minha gratidão pela sólida formação na graduação, que me forneceu os alicerces necessários para alçar voos mais altos. À Universidade Federal do Ceará (UFC) e ao Programa de Pós-Graduação em Engenharia Elétrica e de Computação (PPGEEC), agradeço a oportunidade de me desenvolver como pesquisador e por me permitirem fazer parte de sua história, carregando com orgulho o nome desta renomada instituição.

Ao meu orientador, professor Dr. Wendley Souza da Silva, por toda a sua ajuda, paciência, sugestões e, principalmente, por ter acreditado na realização desta pesquisa e confiar em meus ideais.

Expresso meus sinceros agradecimentos aos professores do PPGEEC-UFC, cujos ensinamentos foram cruciais para minha formação intelectual. Em especial, registro minha gratidão ao professor Márcio André Baima Amora, pela sua didática exemplar, dedicação e pela paciência em compartilhar seu vasto conhecimento. Agradeço também aos ilustres membros da banca examinadora por dedicarem seu precioso tempo à leitura e avaliação deste trabalho, contribuindo com suas valiosas perspectivas.

A jornada dupla entre o trabalho e o mestrado só foi possível graças à compreensão e ao apoio que recebi. Ao meu chefe e amigo, Eliel, minha gratidão pela flexibilidade e paciência, especialmente nos períodos mais desafiadores, permitindo que eu conciliasse minhas responsabilidades com a dedicação que esta dissertação exigia.

Aos colegas do PPGEEC-UFC, agradeço pelo companheirismo, pelas discussões produtivas e, principalmente, pelos sorrisos e momentos de leveza que tornaram a caminhada muito mais agradável. Em especial, agradeço nominalmente aos colegas Alan Freire, Pedro Vasconcelos, Jorge Pedro, Expedito Magalhães e Denner David.

À minha família, meu alicerce e porto seguro, todo o meu amor e gratidão. O incentivo e o carinho de vocês foram o combustível que me moveu. De maneira especial, à minha mãe, Socorro, pelo amor incondicional, pelas orações e por ser o meu maior exemplo de força. Às minhas irmãs, Tatiana e Vanessa, pela amizade, torcida e por serem companheiras

incomparáveis em todos os momentos.

Em especial, um agradecimento que transborda o coração. À minha namorada, Milena, por ser o apoio fundamental ao longo desta etapa tão importante. Seu amor, paciência, incentivo e compreensão nos momentos de ausência foram essenciais para que eu mantivesse o foco e a serenidade. Esta conquista é nossa.

Por fim, agradeço por todas as oportunidades ao longo desses dois anos de mestrado, mesmo as que não deram certo, pois também deram certo. A vida é feita de experiências, tentativas, erros, acertos e recomeços. Cada pedaço desse caminho pavimenta a estrada da vida com momentos únicos. Por isso, ouse tentar, ouse viver, ouse, tente, faça. O maior risco é não correr riscos.

Seja ousado, o universo recompensa os corajosos.

(Geraldo Martins)

RESUMO

A rápida expansão de dispositivos de Internet das Coisas (IoT) e a evolução das ameaças cibernéticas tornam a detecção de ataques de negação de serviço (DoS) e de negação de serviço distribuída (DDoS) uma necessidade crítica para a segurança de sistemas computacionais modernos. Dispositivos de borda, caracterizados por recursos computacionais limitados, apresentam desafios únicos para a implementação de soluções de segurança eficazes, demandando abordagens otimizadas que conciliem eficácia de detecção com viabilidade computacional. Este trabalho apresenta POSEIDDON (Plataforma de Supervisão e Detecção de Intrusões DoS/DDoS Orientada a Nós de Borda), um sistema inovador de detecção de anomalias baseado na análise temporal, especificamente projetado para operar em dispositivos de borda com recursos limitados. A abordagem proposta integra sistematicamente quatro técnicas complementares de análise temporal: entropia de Shannon, ARIMA (AutoRegressive Integrated Moving Average), expoente local de Hölder e médias móveis, para suavização e identificação de tendências. Essa combinação multidimensional permite a modelagem robusta do comportamento normal de tráfego de rede e a detecção eficaz de desvios estatísticos que caracterizam atividades maliciosas. A validação experimental foi realizada utilizando os datasets (CIC-IDS-2017, CSE-CIC-IDS2018, CIC-IDS-2023), com foco específico em ataques de camada de aplicação, particularmente DoS e DDoS Slowloris e SlowHTTPTest, que representam ameaças significativas em razão da capacidade que têm de mimetizar tráfego legítimo. A metodologia experimental incluiu intervalos de confiança de 99%, garantindo rigor estatístico dos resultados. Uma contribuição metodológica importante foi a implementação de validação cruzada entre *datasets* (*cross-dataset*) para avaliação de transferibilidade temporal, demonstrando a capacidade dos modelos de manterem desempenho adequado quando aplicados a dados coletados em períodos distintos. Os resultados experimentais evidenciam o alto desempenho da abordagem proposta, com a Configuração 5 (combinação de médias móveis e entropia de Shannon) alcançando acurácia de 99,9% na validação cruzada temporal. A validação em dispositivo real (Raspberry Pi 3b+) confirmou a viabilidade computacional da solução, demonstrando capacidade de detecção em tempo real, sem comprometer significativamente os recursos computacionais do dispositivo.

Palavras-chave: Detecção de Anomalias; Segurança em IoT; Edge Computing; Ataques DoS/DDoS; Entropia de Shannon; ARIMA; Expoente de Hölder.

ABSTRACT

The rapid expansion of Internet of Things (IoT) devices and the evolution of cyberthreats make the detection of denial-of-service (DoS) and distributed denial-of-service (DDoS) attacks a critical need for the security of modern computing systems. Edge devices, characterized by their limited computing resources, present unique challenges for implementing effective security solutions, requiring optimized approaches that combine detection efficiency with computational solutions. This work presents an Edge-Oriented DoS/DDoS Intrusion Detection and Monitoring Platform, a novel anomaly detection system based on temporal analysis, specifically designed to operate on resource-constrained edge devices. The proposed approach systematically integrates four complementary temporal analysis techniques: Shannon entropy, ARIMA (AutoRegressive Integrated Moving Average), Hölder's local exponent, and moving means, used for smoothing and trend identification. This multidimensional combination enables robust modeling of normal network traffic behavior and effective detection of statistical deviations that characterize malicious activity. Experimental validation was performed using the datasets (CIC-IDS-2017, CSE-CIC-IDS2018, CIC-IDS-2023) with a specific focus on application-layer attacks, notably DoS and DDoS attacks, such as Slowloris and SlowHTTPTest, which pose threats due to their ability to mimic their own traffic. The experimental methodology included 99% confidence intervals, ensuring statistical rigor. A significant methodological contribution was the implementation of cross-dataset cross-validation to assess temporal transferability, demonstrating the models' ability to maintain adequate performance when applied to missing data across different periods. The experimental results demonstrate the high performance of the proposed approach, with Configuration 5 (combination of mobile media and Shannon entropy) achieving 99.9% accuracy in temporal cross-validation. Validation on a real device (Raspberry Pi 3b+) confirmed the solution's computational prediction, demonstrating real-time detection capability without significantly compromising the device's computational resources.

Keywords: ARIMA; DoS/DDoS Attacks; Anomaly Detection; Edge Computing; Shannon Entropy; Hölder Exponent; IoT Security.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ACF	Autocorrelation Function
ARIMA	Autoregressive Integrated Moving Average
ARPANET	Advanced Research Projects Agency Network
AUC	Area under the Curve
BPNN	Backpropagation Neural Network
CNN	Convolutional Neural Network
CoAP	Constrained Application Protocol
DDoS	Distributed Denial of Service
DL	Deep Learning
DNS	Domain Name System
DoS	Denial of Service
DTLS	Datagram Transport Layer Security
GRU	Gated Recurrent Units
HTTP	Hypertext Transfer Protocol
IA	Inteligência Artificial
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IoT	Internet of Things
KNN	K-Nearest-Neighbor
LSTM	Long Short-Term Memory
MIT	Massachusetts Institute of Technology
ML	Machine Learning
MLP	Multilayer Perceptron
MMA	Médias Móveis Aritméticas
MME	Médias Móveis Exponenciais
NTP	Network Time Protocol
PACF	Partial Autocorrelation Function
ROC	Receiver Operating Characteristics
SDN	Software-Defined Network
SVC	Support Vector Classifier
SVM	Support Vector Machine

TCP	Transmission Control Protocol
UDP	User Datagram Protocol

SUMÁRIO

1 INTRODUÇÃO

Em decorrência da evolução da tecnologia da informação e comunicação, o conceito de *Internet of Things* (IoT) foi introduzido, em 1999, por Kevin Ashton, pesquisador do Massachusetts Institute of Technology (MIT) Auto-ID Laboratory (ASHTON *et al.*, 2009). Desde então, a IoT tem experimentado rápido crescimento, sendo implementada em uma ampla variedade de domínios. Para uma referência mais recente, em 2016, havia aproximadamente 4,2 bilhões de dispositivos conectados à internet. Esse número cresceu para estimados 22,2 bilhões em 2025, representando um aumento de 382% (Fortune Business Insights, 2023). A IoT também pode ser entendida como uma rede de tecnologias de última geração, com sensores, atuadores e outros objetos inteligentes interconectados, visando à conveniência e à produtividade. O número de dispositivos conectados aumentou de forma exponencial ao longo da última década, ultrapassando a quantidade de usuários humanos da internet em escala global. Em comparação com 2023, houve um crescimento de 13%, alcançando, em termos globais, aproximadamente 18,8 bilhões de dispositivos (IoT Analytics, 2024).

Entretanto, com a rápida evolução da internet, também se iniciou e evoluiu o cibercrime. No início da difusão da digitalização global, a defesa do mundo digital era mais simples. Além disso, os ataques não eram tão complexos quanto os atuais, mas a tecnologia evoluiu para ambos os lados e os criminosos cibernéticos construíram ferramentas automatizadas para lançar ataques cibernéticos sofisticados ao longo do tempo. Ademais, várias máquinas e plataformas foram adicionadas à internet, incluindo *smartphones*, *tablets*, dispositivos IoT, plataformas em nuvem, plataformas de mídias sociais, entre outros (ASLAN; YILMAZ, 2021). A Tabela 1 apresenta uma classificação do cibercrime ao longo das décadas.

Tabela 1 – Classificação dos cibercrimes ao longo das décadas

Período	Cibercrime
1940s	Os Anos sem crimes cibernéticos
1950s	Década de fraudes telefônicas
1960s	Os Termos de *hacking* e vulnerabilidade aparecem
1970s	Nascimento da segurança da informação
1980s	Os anos da ARPANET para a Internet
1990s	Os vírus e *worms* de computador tornaram-se populares
2000s	A Internet cresce exponencialmente
2010s	Criminosos cibernéticos descobrem diversas falhas de segurança em sistemas de computador
2020s	Os crimes cibernéticos tornaram-se uma indústria

Fonte: Modificado de Aslan *et al.* (2023)

O número e a variedade crescentes de dispositivos IoT fazem com que a superfície de ataque se multiplique muitas vezes. Além disso, a superfície de ataque das redes IoT está aumen-

tando com o aumento da população (número de dispositivos IoT), convolução, heterogeneidade, diversidade, interoperabilidade, portabilidade, mobilidade, localização, topologia e distribuição de objetos (dispositivos, controlador, conectividade, consumidor e serviços). Os seres humanos e o meio ambiente são ameaças à segurança. Por exemplo, catástrofes naturais como tufões, abalos sísmicos, inundações e incêndios são perigos que podem causar sérios danos aos sistemas de computador e redundâncias de *backups*. Assim, o planejamento de contingências é uma excelente forma de proteger infraestruturas estáveis contra esse tipo de ameaça. Já as ameaças humanas são aquelas originadas por ações humanas, podendo ser maliciosas e de natureza interna, quando alguém permite acesso indevido, ou externa, quando indivíduos ou organizações atuam fora da rede, buscando danificar ou interromper o funcionamento do sistema (KRISHNA *et al.*, 2021).

Diante do crescimento exponencial da quantidade de dispositivos conectados à rede e da perspectiva de continuidade desse aumento, observa-se também a evolução de tamanho, complexidade e variabilidade do cibercrime ao redor do mundo, agora valendo-se da vantagem sobre dispositivos pouco seguros frente a tecnologias avançadas de *hacking*. Dentre os vários ataques possíveis, os ataques DoS (*Denial-of-Service*) e DDoS (*Distributed Denial-of-Service*) se destacam nesse meio, principalmente agora, com o uso de dispositivos IoT como mecanismos de ataque. Esse tipo de ataque cibernético existe há mais de 20 anos, e, devido a isso, boa parte das organizações possui algum mecanismo de proteção específico. Entretanto, a evolução da tecnologia e o aumento da quantidade de dispositivos conectados à rede fizeram com que esse tipo de ataque aumentasse drasticamente de tamanho, frequência e complexidade nos últimos anos, tornando necessário o contínuo esforço na detecção e mitigação desses ataques (SYSTEMS, 2023).

Embora as camadas IoT sejam expostas a várias ameaças, algumas são mais propensas a ocorrer e, portanto, mais suscetíveis a ataques. De acordo com (MAHJABIN *et al.*, 2019), os ataques DDoS são prevalentes no meio IoT e se destacam entre os demais ataques cibernéticos. Ressalta-se ainda que, devido à natureza dos dispositivos IoT, esses frequentemente apresentam mais vulnerabilidades, tornando-se alvos fáceis para serem recrutados em *botnets* para realizar ataques DDoS em larga escala. Assim, este trabalho irá se aprofundar em técnicas de detecção e mitigação desse tipo de ataque em camada de aplicação para sistemas IoT, especificamente em servidores de borda. O servidor de borda está mais próximo dos dados, podendo realizar processamento das informações com mais agilidade e enviando a tomada de decisão para um servidor em nuvem, por exemplo. A Figura 2 apresenta um percentual dos ataques mais

prevalentes em computação de borda em sistemas IoT.

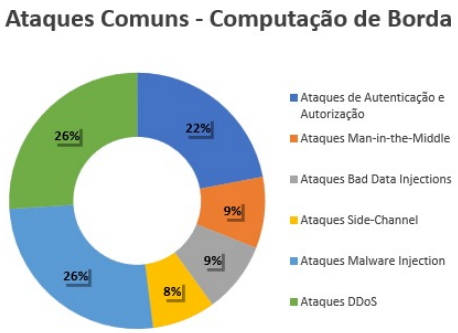

Figura 1 – Porcentagens de ataques mais comuns em computação de borda em sistemas IoT no mundo real.

Fonte: Modificado de Xiao *et al.* (2019).

Como podemos observar na Figura 2, os ataques DDoS são os mais recorrentes em sistemas IoT em escala global, com dados de ataques reais. Assim, com a prerrogativa de otimizar mecanismos de detecção e mitigação desses ataques, visando melhorar a confiabilidade de sistemas IoT, este trabalho irá se aprofundar em técnicas de detecção e mitigação desse tipo de ataque. Além disso, para um melhor entendimento, apresentamos a imagem a seguir, propondo um cenário possível onde se encontra o servidor de borda, sendo representado por um Raspberry PI, onde estariam rodando aplicações e a plataforma de detecção proposta neste trabalho.

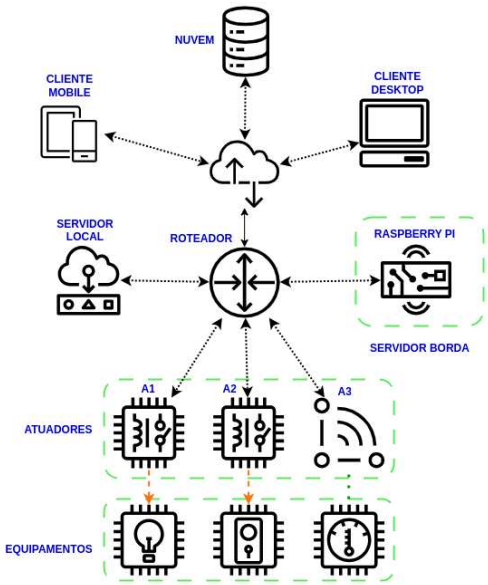

Figura 2 – Topologia proposta para um cenário de detecção com servidor de borda.

Fonte: Próprio autor.

1.1 Questões a serem respondidas

Como questão central da pesquisa, busca-se identificar se é possível desenvolver uma plataforma de detecção de ataques de negação de serviço (DoS/DDoS) de camada de aplicação, especificamente *Slowloris* e *SlowHTTPtest* que seja computacionalmente leve o suficiente para operar de forma eficaz e em tempo real em dispositivos de borda com recursos limitados, mantendo alto desempenho e capacidade de generalização para novos padrões de tráfego.

Para responder à questão principal, foram formuladas as seguintes questões secundárias: Generalização Temporal:

- Como a combinação de diferentes técnicas de análise de séries temporais (entropia, médias móveis, ARIMA, expoente de Hölder) pode melhorar a capacidade de um modelo de *Machine Learning* de generalizar sua detecção para dados de rede coletados em períodos e cenários distintos, sem a necessidade de retreinamento constante?
- Desempenho Computacional: Qual é o impacto no desempenho computacional da implementação de diferentes combinações de técnicas de análise temporal em um dispositivo de borda de baixa capacidade, como um Raspberry Pi?
- Eficácia das Técnicas: Qual combinação de algoritmos de *Machine Learning* e *features* de análise temporal oferece o melhor equilíbrio (*trade-off*) entre precisão na detecção, robustez contra novos tipos de ataque e eficiência computacional?
- Robustez do Modelo: A fusão de múltiplos *datasets* de diferentes épocas e topologias de rede pode resultar em um modelo de detecção mais robusto e atemporal, capaz de identificar uma gama mais ampla de ataques de camada de aplicação de alto desempenho?

1.2 Desafios e Justificativa

Diante da evolução do cibercrime e a necessidade de maior confiabilidade de sistemas IoT, surgem alguns desafios que embasam a pesquisa realizada neste trabalho.

1.2.1 Desafios da Pesquisa

Tendo em vista os ataques DoS e DDoS e suas categorias, os ataques de camada de aplicação são mais difíceis de detectar e mitigar devido ao seu tráfego se assemelhar bastante ao tráfego real de rede. Diante disso, buscam-se formas e modelos de detecção e mitigação que possam se adaptar melhor às mudanças e evoluções temporais em cenários diversos desses

ataques ao longo do tempo, principalmente em servidores de borda de baixo poder computacional.

1.2.2 Justificativa

A relevância desta pesquisa fundamenta-se em três pilares principais: primeiro, a crescente importância dos ambientes IoT e *edge computing* na infraestrutura digital moderna, onde a segurança cibernética representa um desafio crítico. Segundo, a necessidade específica de métodos de detecção eficientes para ataques de baixo volume que podem passar despercebidos por sistemas tradicionais, diante da maior dificuldade de detecção devido à sua semelhança com o tráfego benigno e à sofisticação dos ataques modernos. Terceiro, a oportunidade de aplicar técnicas avançadas de análise temporal, entropia combinada com algoritmos de *Machine Learning* (ML) para melhorar o desempenho e a robustez da detecção.

A escolha por focar nos ataques de camada de aplicação *Slowloris* e *SlowHTTPTest* justifica-se pela sua prevalência em ambientes IoT e pela dificuldade inerente à detecção devido ao seu comportamento de baixo volume e longa duração. Esses ataques exploram vulnerabilidades específicas de protocolos de aplicação e podem ser particularmente devastadores em dispositivos com recursos limitados.

1.3 Contribuições

Nesta seção, serão elencadas as contribuições realizadas no trabalho de dissertação, sendo divididas em contribuições metodológicas, científicas e práticas, com o objetivo de enfatizar, de forma detalhada, o trabalho desenvolvido.

1.3.1 Contribuições Metodológicas

Nas contribuições metodológicas, realizamos uma combinação de modelagens que auxiliaram na obtenção dos resultados finais apresentados. Foi realizada uma investigação aprofundada de técnicas e métodos para extrair uma validação temporal dos modelos aplicados, implementando técnicas de análises temporais com características distintas, mas que possibilitassem melhores resultados. A forma como cada técnica foi aplicada a cada tipo de dado foi fundamental para o êxito dos resultados. Identificar e aplicar técnicas em segmentos de *features*, como extrair informações numéricas contínuas de *features* binárias com médias móveis, entropia, expoente de Hölder e ARIMA. Além disso, a entropia de Shannon aplicada diretamente a *flags*

puramente binárias gerou resultados impactantes, principalmente com as *flags* PSH *Flag* e ACK *flag*, nas quais as *flags* de cabeçalho do protocolo Transmission Control Protocol (TCP) garantem o recebimento dos dados com ACK, enquanto a PSH lida com urgência e latência.

1.3.1.1 Plataforma de Supervisão e Exposição de Intrusões DoS/DDoS Orientada a Nós de Borda

A plataforma desenvolvida possui grande relevância e contribuição metodológica, devido à sua validação temporal e ao desempenho computacionalmente leve, possibilitando aplicações no mundo real em dispositivos de borda simples, como um Raspberry PI 3b+, e ainda sem utilizar toda a sua capacidade computacional. Além disso, a eficácia comprovada, agora com o retreinamento em um cenário de *cross-dataseting* torna a solução robusta e atemporal, ampliando também o leque de ataques de camada de aplicação da plataforma. Trata-se de uma plataforma *end-to-end*, ou seja, é realizada desde a engenharia de *features* até a inferência do modelo, classificando os ataques. Adicionalmente, avaliamos o desempenho de diversas técnicas de análises temporais combinadas, buscando métricas que possibilitem uma aplicação real com desempenho computacionalmente leve para dispositivos de borda.

1.3.1.2 Protocolo de Validação e Transferibilidade Temporal

Um dos grandes desafios da pesquisa foi obter um modelo de ML que perpassasse a barreira temporal com alto desempenho, buscando modelos que possam ser implementados no mundo real. Para isso, realizamos várias configurações distintas, utilizando técnicas variadas que, nos testes com o mesmo cenário em que os dados foram treinados, apresentaram resultados excelentes. Contudo, a validação temporal mostrou que nem todos os algoritmos e configurações foram capazes de generalizar para tráfego de dados em cenários distintos. Apesar disso, conseguimos obter êxito ao encontrar uma configuração leve, capaz de proporcionar uma validação temporal confiável para aplicações reais.

1.3.2 Contribuições Científicas

1.3.2.1 Análise Comparativa de Técnicas de Análise Temporal

Na realização do experimento deste trabalho, buscou-se avaliar o comportamento do desempenho temporal dos modelos de ML com a aplicação de técnicas de análise de séries

temporais, em especial o expoente local de Hölder e a análise de entropia de Shannon, combinadas a outras técnicas de suavização e predição para séries temporais. Para os dados do mesmo cenário, todas as configurações possibilitaram excelentes resultados, mostrando que os modelos generalizaram bem para o contexto de dados em que foram treinados. Entretanto, sabemos que o tráfego de rede pode ter comportamento anômalo distinto e muito volátil, o que nos força a buscar modelos que generalizem bem para cenários de tráfego de rede ainda não conhecidos, detectando e classificando anomalias na rede, realizando retreinamento *off-line*.

1.3.2.2 Caracterização de Padrões Temporais de Ataques

Diante do experimento realizado, constatou-se que a suavização dos dados por meio de médias móveis e entropia de Shannon apresentou mais resultados positivos na validação temporal de cenários. No entanto, evidenciou-se também que os modelos combinados, como ARIMA com MLP e Hölder com AdaBoost, demonstraram alto desempenho. Esse excelente resultado se deve à capacidade da entropia de Shannon de lidar com a aleatoriedade da informação, sendo ideal para quantificar a "surpresa", uma vez que eventos menos prováveis carregam mais informação, elevando a entropia. Sua fórmula matemática calcula a quantidade média de informação (bits) necessária para representar os eventos gerados por uma fonte. Assim, a combinação dessas técnicas, aliada à metodologia abordada mostrou excelentes resultados.

1.3.3 Contribuições Práticas e Aplicabilidade em Ambientes IoT

Reafirmando, os ataques de camada de aplicação são de difícil detecção, uma vez que seu tráfego de ataque se assemelha bastante ao tráfego real. Contudo, para a pesquisa em questão, esse não é o único desafio, pois a proposta da pesquisa é que apliquemos esta plataforma em dispositivos de borda com poucas capacidades computacionais e que obtivemos êxito ao implementarmos com sucesso em um Raspberry PI 3b+, ainda com capacidade de processamento para implementação de outras aplicações no dispositivo. Além disso, sem perder desempenho quanto ao seu funcionamento nesses ambientes limitados.

1.4 Organização do Trabalho

O trabalho é organizado em capítulos seguindo um escopo lógico da proposta realizada na pesquisa. O Capítulo 2 traz os objetivos gerais e específicos da pesquisa, definindo

o direcionamento do estudo. Em seguida, o capítulo 3 apresenta toda a fundamentação teórica, abordando conceitos fundamentais dos ataques DoS e DDoS, mais especificamente os de camada de aplicação, seguidos das técnicas de séries temporais utilizadas na metodologia e suas principais características, as métricas usadas na validação dos modelos e os algoritmos de ML adotados.

O capítulo 4 traz uma vasta revisão da literatura elencando métodos e formas de detecção comerciais e pesquisas relacionadas, prevenção e mitigação de ataques DoS/DDoS, comparando categorias de ataques e técnicas de detecção e mitigação usando IA.

A metodologia utilizada, apresentada no capítulo 5, inicia da visão experimental e são mostrados os *datasets* analisados e os que foram utilizados na pesquisa. Após isso, o detalhamento das configurações utilizadas no experimento é realizado, seguido de fluxogramas com o passo a passo da abordagem e os detalhes dos conjuntos finais de *features* e hiperparâmetros dos modelos de ML utilizados.

No capítulo 6, são apresentados os resultados obtidos a partir do estudo realizado por meio de uma análise quantitativa dos resultados, evidenciando as melhores configurações e modelos para aplicações em servidores de borda com baixo poder computacional. Além disso, uma avaliação do desempenho computacional é realizada para viabilizar implementações reais em dispositivos de borda.

Por fim, o capítulo 7 apresenta as conclusões diante da pesquisa realizada, evidenciando detalhes do estudo como a metodologia utilizada, a combinação de técnicas e os resultados obtidos, finalizando com a perspectiva de trabalhos futuros.

2 OBJETIVOS DA PESQUISA

2.1 Objetivo Geral

Desenvolver uma plataforma de ML para detectar ataques de baixo volume DoS e DDoS de camada de aplicação em tempo real, visando o aumento de confiabilidade de sistemas IoT na borda para sistemas IoT.

2.2 Objetivos Específicos

- Adotar uma taxonomia consolidada na literatura;
- Investigar técnicas de análises temporais que proporcionem melhor generalização de dados de tráfego de redes para validações temporais;
- Implementar combinações de técnicas de análises temporais para detecção de anomalias em tráfego de redes IoT;
- Realizar validação cruzada entre *datasets* para validação temporal e avaliar generalização dos modelos;
- Analisar o desempenho computacional dos modelos nos cenários e configurações para aplicações em dispositivos de borda;
- Aplicar experimentos de melhor desempenho com *dataset* híbrido e re-treinar modelos de melhor desempenho para expandir as classes de ataques e atualizar os modelos.

3 FUNDAMENTAÇÃO TEÓRICA

3.1 Ataques DoS e DDoS

Ataques DoS são um tipo de ataque cibernético que é articulado para derrubar ou interromper serviços de internet originados de um único atacante para um único alvo. Já os ataques DDoS, em sua forma distribuída, utilizam um grande volume de atacantes (*bots* ou zumbis) para atacar um único alvo, tornando-os mais eficazes. Todavia, sites ou aplicativos móveis tornam-se indisponíveis para seus usuários, prejudicando a reputação das empresas ou organizações. O ataque DoS, em sua forma distribuída, opera da mesma maneira, porém com maior poder de ataque (GOEL *et al.*, 2023). A Figura 3 mostra um exemplo com a diferença entre o tráfego normal e os ataques DoS e DDoS, onde em Figura 3 (a) mostra o tráfego normal, em Figura 3 (b) o ataque DoS representado por requisições incompletas feitas por apenas um *host* e, finalmente, Figura 3 (c) que recebe múltiplas requisições HTTP incompletas de vários *hosts* distintos.

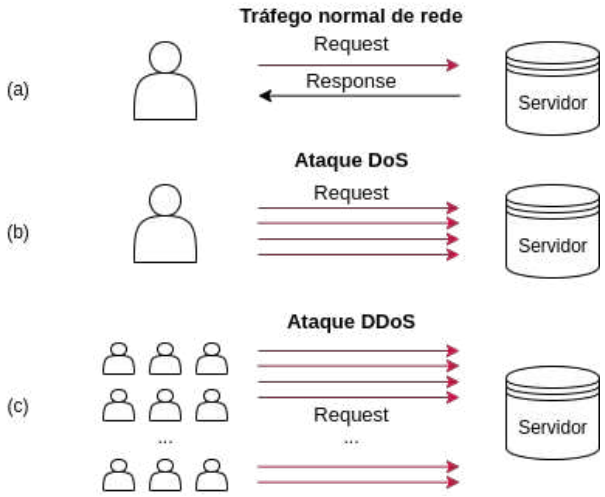

Figura 3 – Diferença entre tráfego normal, ataques DoS e DDoS via protocolo HTTP.
Fonte: Próprio autor.

Esses ataques podem ocorrer de várias formas. Uma maneira de atacar é gerar um enorme tráfego de rede e transmitir solicitações volumosas para a vítima. Com isso, podem causar repercussões financeiras significativas para grandes empresas. O objetivo principal dos ataques DoS e DDoS é interromper os serviços da rede por meio de atividades maliciosas, como gerar tráfego excessivo e irrelevante, sobrecarregar redes com solicitações excessivas e esgotar recursos vitais, como largura de banda e *hardwares*, tornando dispositivos, softwares, serviços de rede e recursos indisponíveis para os consumidores-alvo. Como resultado, esses ataques

geralmente levam a consequências adicionais, como superaquecimento e danos à propriedade (UDDIN *et al.*, 2024; DAMGHANI *et al.*, 2019; TSIKNAS *et al.*, 2021; KRISHNA *et al.*, 2021; SENGUPTA *et al.*, 2020; SASI *et al.*, 2024; KHANAM *et al.*, 2020; AI *et al.*, 2020; HASSIJA *et al.*, 2019; CHEN *et al.*, 2018).

Para uma compreensão mais detalhada e aprofundada da evolução dos ataques DDoS, a próxima seção tratará exclusivamente da taxonomia, das categorias de ataques, dos métodos e das contramedidas.

3.1.1 Taxonomia dos ataques DoS e DDoS

Os ataques DoS e DDoS são divididos em três categorias: ataques volumétricos, ataques baseados em protocolos e ataques de camada de aplicação. A seguir, cada uma dessas categorias será detalhada.

- **Ataques volumétricos:** o objetivo principal desse tipo de ataque é sobrecarregar a largura de banda da rede ou os recursos de infraestrutura da vítima (servidor, roteador ou rede). Esses ataques enviam um massivo volume de dados de tráfego para saturar os canais de comunicação, impedindo o processamento do tráfego legítimo. Os atacantes utilizam *botnets* ou servidores mal configurados (como Domain Name System (DNS) ou Network Time Protocol (NTP)) para gerar tráfego massivo direcionado à vítima. Esse tráfego é, muitas vezes, amplificado, fazendo com que uma pequena solicitação gere uma resposta significativamente maior. Para isso, são utilizados protocolos que permitem comunicação de alto volume, como User Datagram Protocol (UDP), Internet Control Message Protocol (ICMP), DNS e NTP (UDDIN *et al.*, 2024; ALAHMADI *et al.*, 2023; PRASAD; CHANDRA, 2022; BALA; BEHAL, 2024; MISHRA; PANDYA, 2021).
- **Ataques baseados em protocolo:** os ataques baseados em protocolos exploram falhas ou limitações inerentes aos protocolos de comunicação da internet, como TCP, UDP, ICMP, DNS e outros. O objetivo desses ataques é sobrecarregar os recursos de uma rede ou sistema, como tabelas de conexão, largura de banda e processamento de pacotes. Para isso, utilizam mecanismos como o envio de pacotes maliciosos, que consomem recursos alocados para conexões legítimas. Em outros casos, enviam pacotes em alta frequência, às vezes amplificados, com o intuito de consumir a largura de banda de um servidor, causando sobrecarga (UDDIN *et al.*, 2024; ALAHMADI *et al.*, 2023; BALA; BEHAL, 2024; MISHRA; PANDYA, 2021).

- **Ataques de Camada de Aplicação:** os ataques de camada de aplicação constituem uma categoria de ataque DDoS cujo objetivo é sobrecarregar os recursos de um serviço ou aplicação específica, explorando a lógica de funcionamento da aplicação, em vez de depender apenas do volume de tráfego. Em geral, esses ataques imitam o comportamento de tráfego legítimo, tornando-os difíceis de detectar e mitigar (UDDIN *et al.*, 2024; ALAHMADI *et al.*, 2023; BALA; BEHAL, 2024; MISHRA; PANDYA, 2021). Alguns dos ataques mais comuns na camada de aplicação são descritos no tópico a seguir.

A seguir, serão abordadas as categorias dos ataques DoS e DDoS, bem como os tipos mais comuns de ataques em cada uma delas. Dessa forma, todos os ataques serão apresentados dentro da taxonomia de ataques DDoS em sistemas IoT na Figura 4.

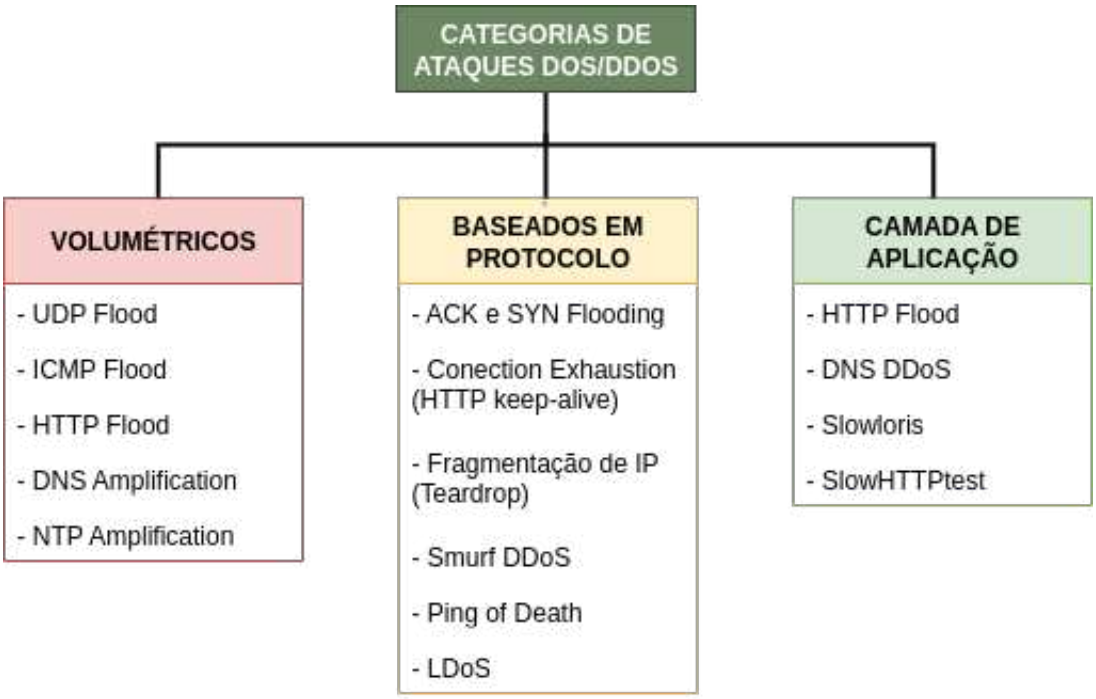

Figura 4 – Taxonomia das categorias e ataques DoS/DDoS em sistemas IoT.
Fonte: Autor.

3.1.2 Ataques de Camada de Aplicação

Os ataques de camada de aplicação, foco deste trabalho, abordarão dois ataques: *Slowloris* e o *SlowHTTPtest*, por serem os representantes dessa categoria disponíveis nos *datasets* utilizados nesta pesquisa.

Ataques DoS e DDoS *Slowloris*: o ataque slowloris é um tipo de ataque de camada de aplicação que envolve o uso de solicitações *Hypertext Transfer Protocol (HTTP)* parciais. A principal

diferença reside na sua abordagem. O Slowloris concentra-se em manter as conexões com o servidor abertas pelo maior tempo possível. Para isso, ele envia requisições HTTP com cabeçalhos parciais e nunca as completa. O servidor, por sua vez, aguarda o restante dos dados, mantendo a conexão ativa e alocando recursos para ela. Ao multiplicar essas conexões "semiabertas", o Slowloris exaure o limite de conexões concorrentes do servidor, impedindo que usuários legítimos consigam acessá-lo (WIENS; ZITZMANN, 1999; UDDIN *et al.*, 2024).

Ataques DoS e DDoS SlowHTTPtest: são aplicados por meio de uma ferramenta mais versátil, capaz de simular diferentes tipos de ataques lentos, incluindo uma modalidade semelhante à do Slowloris. Suas principais variações são:

- Slowloris mode: opera de forma similar ao ataque original, enviando cabeçalhos HTTP de forma lenta e fragmentada;
- Slow HTTP POST: neste modo, o ataque envia um cabeçalho HTTP POST completo, que indica o envio de um corpo de dados. No entanto, o envio desses dados é feito de maneira extremamente lenta, mantendo a conexão ocupada e consumindo os recursos do servidor, que aguardam o término da transmissão.
- Slow Read: o atacante envia uma requisição HTTP legítima, mas lê a resposta do servidor de forma deliberadamente vagarosa. Isso obriga o servidor a manter os dados no buffer de envio por um longo período, consumindo memória e limitando sua capacidade de atender a outras solicitações.

3.2 Técnicas de Séries Temporais para Análise de Tráfego

Alguns dos modelos utilizados para descrever séries temporais são processos estocásticos, em que não se tem uma característica determinística, isto é, controlados por leis probabilísticas. A construção dos modelos para descrever séries temporais depende de vários fatores, entre eles, o comportamento do fenômeno ou o conhecimento do objeto de análise. A seguir, falaremos de algumas técnicas de séries temporais aplicadas aos dados de tráfego de rede em busca de modelos que permeiem o tempo e generalizem com eficácia tráfego de rede para detecção de ataques DoS e DDoS. Antes de tudo, para analisar uma série temporal, é necessário determinar qual é o tipo de classe de processo estocástico (MORETTIN; TOLOI, 2018). A seguir são detalhadas algumas técnicas utilizadas na pesquisa.

3.2.1 Médias Móveis Aritméticas e Médias Móveis Exponenciais para Suavização de Dados

As médias móveis foram as primeiras ferramentas utilizadas para suavização de dados, retirando ruídos apresentados pelas oscilações e indicando de forma confiável a tendência de uma série. Existem diversos tipos de médias móveis, entretanto, neste trabalho, serão utilizadas as mais populares: as Médias Móveis Aritméticas (MMA) e Médias Móveis Exponenciais (MME) (MORETTIN; TOLOI, 2018). A MMA atribui peso igual a todos os períodos, sendo representada na fórmula a seguir, onde $V1+V2+..V_n$ são os valores aferidos dentro da janela determinada n.

$$MMA = \frac{V1+V2+...+V_n}{n} \tag{3.1}$$

Já a MME tem proporcionalidade de peso diferente, como mostra a fórmula a seguir:

$$MME = V_{atual} + K + MME_{anterior} * (1-K) \tag{3.2}$$

Onde o V_{atual} é o valor atual, K é o fator de ponderação ou constante de suavização sendo determinado pelo tamanho da janela n usada para a média móvel, que é determinada pela equação a seguir e que sempre será um valor entre 0 e 1.

$$K = \frac{2}{n+1} \tag{3.3}$$

Contudo, foram utilizadas as funções *rolling* e *ewm* da biblioteca pandas no *python* para MMA e MME, respectivamente.

3.2.2 Modelos ARIMA

O modelo Autoregressive Integrated Moving Average (ARIMA) é uma classe de modelos estatísticos que descreve uma série temporal univariada sendo então, uma única variável ao longo do tempo. Para isso, ela usa os próprios valores passados, a sigla ARIMA representa três componentes principais do modelo, são eles:

- **AR (AutoRegressivo)**: Indica que o valor atual da série temporal é uma função linear de seus valores passados. O componente autoregressivo de ordem (p), denotado por $AR(p)$), significa que o valor atual y_t depende dos (p) valores anteriores da série $y_{(t-1)}, y_{(t-2)} \cdots, y_{(t-p)}$.
- **I (Integrado):** Refere-se ao processo de diferenciação necessário para tornar a série temporal estacionária. Uma série é estacionária quando suas propriedades estatísticas (média, variância e autocorrelação) não mudam ao longo do tempo. O componente

integrado de ordem (d), denotado por $I(d)$, representa o número de vezes que a série precisa ser diferenciada para se tornar estacionária.

- **MA (Médias Móveis):** Indica que o valor atual da série temporal é uma função linear de erros de previsão passados (termos de erro aleatórios). O componente de médias móveis de ordem (q), denotado por $MA_{(q)}$, significa que o valor atual y_t depende dos (q) erros de previsão anteriores $(\varepsilon_{t-1}, \varepsilon_{t-2}, ..., \varepsilon_{t-q})$, em que ε representa o erro calculado nas previsões anteriores.

Um modelo ARIMA é, portanto, especificado por três parâmetros: ((p, d, q)). A escolha desses parâmetros é um passo crítico na modelagem ARIMA e geralmente envolve a análise das Autocorrelation Function (ACF) e Partial Autocorrelation Function (PACF) da série temporal (CHATFIELD, 2003).

Teste de Estacionariedade e Diferenciação (Parâmetro (d)): Para a aplicação de modelos ARIMA, um pré-requisito fundamental é a estacionariedade, uma vez que séries temporais que exibem tendências ou sazonalidade precisam ser transformadas em séries estacionárias através do processo de diferenciação. A diferenciação de ordem (d) envolve subtrair o valor da série no tempo (t-d) do valor no tempo (t). Por exemplo, uma diferenciação de primeira ordem ($d = 1$) remove uma tendência linear, enquanto uma diferenciação sazonal pode remover padrões repetitivos ao longo de um período fixo (TSAY, 2005). No contexto de detecção de ataques, a estacionariedade do tráfego de rede normal é um conceito importante. Desvios da estacionariedade podem indicar a presença de anomalias ou ataques. A aplicação de diferenciação pode ajudar a isolar esses desvios, tornando-os mais evidentes para o modelo.

Componente Autoregressivo (p): O componente $AR(p)$ modela a dependência entre uma observação e um certo número de observações passadas. Isso significa que o valor atual da série é uma combinação linear dos (p) valores anteriores. Em termos de tráfego de rede, isso implica que o volume de tráfego atual pode ser influenciado pelos volumes de tráfego das últimas (p) unidades de tempo. A ordem (p) é determinada pela análise da Função de Autocorrelação Parcial (PACF), que mede a correlação entre uma observação e uma observação anterior, removendo a influência das observações intermediárias (CHATFIELD, 2003).

Componente de Médias Móveis (Parâmetro (q)): O componente $MA(q)$ modela a dependência entre uma observação e um termo de erro residual de um certo número de observações

passadas. Isso significa que o valor atual da série é influenciado pelos erros de previsão das últimas (q) unidades de tempo. A ordem (q) é determinada pela análise da Função de Autocorrelação (ACF), que mede a correlação entre uma observação e uma observação anterior, incluindo a influência das observações intermediárias (CHATFIELD, 2003).

3.2.3 Entropia de Shannon e Análise de Informação

A entropia tem origem na termodinâmica, sendo definida como a energia térmica que não pode ser convertida em trabalho mecânico. Contudo, Shannon adaptou esse conceito para a comunicação, identificando a entropia como a medida da incerteza ou da quantidade média de informação contida em uma variável aleatória ou fonte de dados. Essa medida é crucial para otimizar processos de transmissão, armazenamento e processamento de dados. Silva (2018) detalha o conceito de entropia na teoria da informação, destacando que ela quantifica a desordem ou a incerteza de uma fonte de dados, sendo máxima quando as probabilidades dos estados são iguais. Neste trabalho, a entropia de Shannon foi aplicada em todas as *features* binárias com maior correlação de dados. Além disso, é importante ressaltar que o uso da entropia de Shannon em variáveis binárias é especialmente útil, pois avalia a incerteza ou a impureza de variáveis binárias, sendo a entropia mais intuitiva para esses dados. Para variáveis contínuas, a discretização pode ser necessária antes de calcular a entropia. Para uma variável aleatória X, com possíveis resultados x_i e probabilidades estimadas a partir dos valores distintos dentro da janela são denominadas por p_i, onde a entropia é dada por:

$$H(X) = -\sum_i p_i \log_2(p_i) \tag{3.4}$$

Se a variável for um vetor de observações $x = \{x_1, x_2, \ldots, x_n\}$, que chamaremos de janela, onde n é o número total de observações, o procedimento comum é:

1. Contar a frequência de cada valor distinto x_i.
2. Calcular a probabilidade de cada valor:

$$p_i = \frac{\text{contagem de } x_i}{n} \tag{3.5}$$

3. Calcular a entropia:

$$H(X) = -\sum_i p_i \log_2(p_i) \tag{3.6}$$

3.2.4 Expoente Local de Hölder

A dimensão fractal de uma série temporal mede a complexidade de um sinal. A ideia básica pode ser entendida como a área habitada pela série temporal em um plano bidimensional, ou seja, a série temporal sendo colocada em uma grade de espaçamento igual e verificando o número de caixas de grade necessárias para cobri-la. Portanto, a dimensão fractal é a proporção das caixas que cobrem a série temporal e a área geral do gráfico. Esse processo é conhecido como contagem de caixas. Também foi caracterizado como uma medida da capacidade de preenchimento de espaço de uma série temporal que informa como um fractal escala de forma diferente do espaço em que está inserido. Uma dimensão fractal não precisa ser um número inteiro. A dimensão fractal de uma série temporal auto-afim pode ter valores $D = 2 - H$, onde D é a dimensão fractal e H o expoente de Hurst. De acordo com Raubitzek e Neubauer (2021), podemos calcular o expoente local de Hölder aplicando as equações abaixo.

Dado um sinal $\mathbf{x} = [x_1, x_2, \ldots, x_n]$, primeiro obtemos as amplitudes normalizadas correspondentes como:

$$\hat{x}_i = \frac{x_i}{\sqrt{\sum_{i=1}^{n} x_i^2}}, \quad \text{tal que} \quad \sum_{i=1}^{n} \hat{x}_i^2 = 1. \tag{3.7}$$

Onde $\hat{x}$ representa a série temporal normalizada, i é o índice de contagem dos pontos da série e n é o total de amostras da mesma. Em seguida, muito semelhante à análise R/S feita no expoente de Hurst, definimos janelas (ou períodos) como $(t - \tau, t)$, onde τ é o comprimento da janela e t é o ponto de análise local. Desta forma, calculamos as energias do sinal correspondente como:

$$E_{t,\tau} = \sum_{j=t-\tau}^{t} \hat{x}_j^2. \tag{3.8}$$

Finalmente, estimamos o expoente local de Hölder (ou singularidade) como:

$$\alpha_t(\tau) = \frac{\log\left(E_{t,\tau}\right)}{\log(\tau)}. \tag{3.9}$$

Em que, semelhante ao expoente de Hurst ou dimensão fractal, o expoente α quantifica o grau de regularidade ou irregularidade (singularidade) de uma função em um ponto t, para diferentes comprimentos de janela τ, normalmente calculados por meio de análise de regressão, representando o próprio expoente de Hölder. Para um processo monofractal, α_t permanece constante para todos os t, enquanto para processos multifractais, α varia de ponto a ponto, fornecendo uma medida do comportamento fractal dos dados da série temporal.

3.3 Métricas de Avaliação

Neste trabalho, a avaliação dos modelos de detecção de intrusão vai além de uma única métrica. Devido à natureza dos dados de tráfego de rede, caracterizados por um severo desbalanceamento de classes (onde ataques são eventos raros), e aos requisitos de implementação em dispositivos de borda, uma abordagem de avaliação holística é fundamental. As métricas foram selecionadas para avaliar os modelos sob as perspectivas de eficácia, robustez e eficiência computacional.

3.3.1 Matriz de Confusão

A Matriz de Confusão é a ferramenta fundamental para visualizar e entender o desempenho de um algoritmo de classificação. Ela sumariza o número de previsões corretas e incorretas feitas por um modelo, categorizando-as em relação às classes reais. Para um problema de classificação binária (duas classes, 'Normal' e 'Ataque'), a matriz é composta por quatro elementos principais:

- **Verdadeiros Positivos (VP)**: Amostras que são da classe positiva ('Ataque') e foram corretamente classificadas como positivas.
- **Verdadeiros Negativos (VN)**: Amostras que são da classe negativa ('Normal') e foram corretamente classificadas como negativas.
- **Falsos Positivos (FP)**: Amostras que são da classe negativa ('Normal'), mas foram incorretamente classificadas como positivas (erro Tipo I).
- **Falsos Negativos (FN)**: Amostras que são da classe positiva ('Ataque'), mas foram incorretamente classificadas como negativas (erro Tipo II).

Em problemas multiclasse, a matriz de confusão se expande para uma tabela $N * N$, onde N é o número de classes e cada célula (i, j) representa a quantidade de amostras da classe real i que foram classificadas como classe j. A diagonal principal da matriz representa as classificações corretas como mostra a Figura 5 (SATHYANARAYANAN; TANTRI, 2024).

Acurácia: A acurácia é a medida mais usada em problemas de classificação e indica a proporção com que o classificador acertou. O valor da métrica é dada por,

$$\text{Acurácia} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.10}$$

onde TP é o número de verdadeiros positivos, TN é o número de verdadeiros negativos, FP o número de falsos positivos e FN o número de falsos negativos.

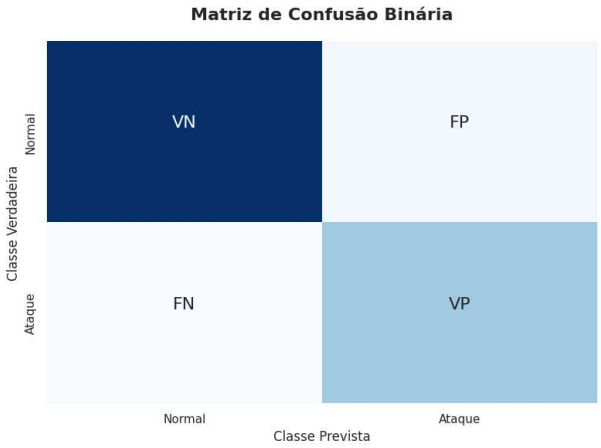

Figura 5 – Modelo de matriz de confusão.
Fonte: Próprio autor.

Precisão: A precisão estima o quanto podemos confiar num modelo quando ele prevê que um exemplo pertence a uma determinada classe. A precisão é a razão entre o número de verdadeiros positivos e o número total de previsões positivas feitas pelo modelo (verdadeiros positivos e falsos positivos), ou mais formalmente,

$$\text{Precisão} = \frac{TP}{TP+FP} \tag{3.11}$$

Sensibilidade: Já a sensibilidade é o número de exemplos que o modelo identificou corretamente como sendo da classe positiva dividido pelo número total de exemplos que realmente pertencem a essa classe. Mais formalmente, a sensibilidade é dada por,

$$\text{Sensibilidade} = \frac{TP}{TP+FN} \tag{3.12}$$

onde FN é o número de falsos negativos.

F1-Score: O F1 score é a média harmônica entre a precisão e a sensibilidade, e mede o equilíbrio entre as duas métricas. Matematicamente, a métrica é dada por,

$$\text{F1} = 2 * \frac{Preciso * Sensibilidade}{Preciso + Sensibilidade} \tag{3.13}$$

3.3.2 As métricas Receiver Operating Characteristic (ROC) e Area under the Curve (AUC)

Enquanto métricas como Precisão e Recall fornecem um resultado instantâneo do desempenho do modelo em um determinado ponto de corte de classificação, a Receiver Operating Curve (ROC) e a Area under the Curve (AUC) oferecem uma visão mais abrangente da capacidade de discriminação de um classificador binário em diferentes limiares de decisão. Elas são particularmente úteis quando se deseja avaliar o *trade-off* entre a taxa de verdadeiros positivos e a taxa de falsos positivos.

A AUC é uma métrica excelente para comparar diferentes modelos de detecção de ataques em IoT, especialmente quando a proporção de ataques é muito menor que a de tráfego normal. Um modelo com alta AUC indica uma boa capacidade de distinguir entre tráfego benigno e malicioso, independentemente do ponto de corte escolhido para a decisão final (SATHYANARAYANAN; TANTRI, 2024).

Curva ROC: A Curva ROC é um gráfico que plota a Taxa de Verdadeiros Positivos (TVP, ou Recall) contra a Taxa de Falsos Positivos (TFP) em vários limiares de classificação como apresenta a Figura 6.

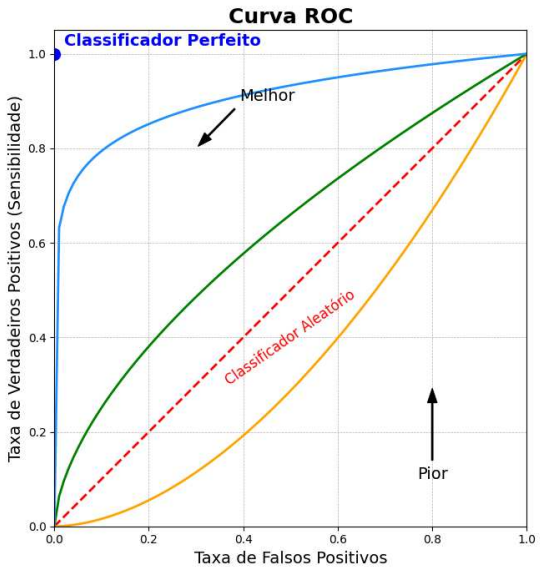

Figura 6 – Exemplo de curva ROC.
Fonte: Adaptado de Sathyanarayanan e Tantri (2024).

A TFP é calculada como:

$$TFP = \frac{FP}{FP + VN} \tag{3.14}$$

Cada ponto na Curva ROC representa um par (TFP, TVP) para um determinado limiar de decisão. Ao variar o limiar de 0 a 1, diferentes pontos são gerados, formando a curva. Um classificador ideal teria uma curva que passa pelo canto superior esquerdo do gráfico (TFP=0, TVP=1), indicando que ele pode alcançar 100

Área Sob a Curva AUC A Área Sob a Curva ROC (AUC-ROC) é uma métrica escalar que quantifica o desempenho geral de um classificador. Ela representa a probabilidade de que o modelo classifique uma amostra positiva aleatoriamente escolhida mais alto do que uma amostra negativa aleatoriamente escolhida. O valor da AUC varia de 0 a 1:

- AUC = 1: Classificador perfeito;

- AUC = 0,5: Classificador aleatório;
- AUC < 0,5: Classificador inferior ao aleatório.

3.3.3 Intervalo de Confiança

Ao avaliar o desempenho de um modelo de ML, as métricas como acurácia, precisão, recall e F1-score são frequentemente apresentadas como valores pontuais. No entanto, esses valores são estimativas baseadas em um conjunto de dados específico (o conjunto de teste ou de validação). Para compreender a verdadeira confiabilidade dessas estimativas e como elas podem variar se o modelo fosse aplicado a diferentes amostras de dados, utilizamos o conceito de Intervalo de Confiança (IC).

Um Intervalo de Confiança é um *range* de valores, calculado a partir dos dados de uma amostra, que provavelmente contém o verdadeiro valor de um parâmetro populacional (a verdadeira acurácia do modelo). Um IC é sempre associado a um nível de confiança (95%, 99%), que indica a probabilidade de que o intervalo contenha o verdadeiro parâmetro se o processo de amostragem e cálculo fosse repetido muitas vezes. Por exemplo, um IC de 95% para a acurácia de um modelo significa que, se o experimento de treinamento e teste for repetido muitas vezes, 95% dos intervalos construídos dessa forma conterão a verdadeira acurácia do modelo Kim e Jeong (2015).

Para métricas de desempenho de modelos de ML, especialmente quando se utiliza validação cruzada (como *K-Fold* ou *Stratified K-Fold*), calculamos a métrica (acurácia) para cada *fold*. Isso nos dá uma distribuição de valores para a métrica. O intervalo de confiança é então construído em torno da média desses valores, utilizando métodos estatísticos como o *t-Student* (para amostras pequenas) ou a aproximação normal (para amostras grandes).

3.3.4 Desempenho Computacional

Em sistemas de *Machine Learning* implantados em ambientes de IoT e *Edge Computing*, o desempenho computacional transcende a mera otimização de recursos, é uma métrica de avaliação tão crítica quanto qualquer outra. Dispositivos de borda, como Raspberry PI ou ATLAS da Huawei, operam com restrições significativas de processamento, memória, energia e largura de banda. Nesses cenários, um modelo que é altamente preciso, mas computacionalmente intensivo, pode ser inviável. A capacidade de um modelo de processar dados e gerar uma detecção em tempo hábil é frequentemente denominada latência ou tempo de detecção.

Para uma avaliação do modelo mais eficaz computacionalmente, iremos aferir o tempo de execução de algumas etapas do *pipeline*, são elas:

- **Engenharia de *features*:** O tempo gasto na transformação dos dados brutos em *features* significativas para o modelo. Isso pode incluir cálculos de médias móveis, entropia, extração de estatísticas, etc. Em um ambiente de inferência em tempo real, esta etapa é crucial, pois precisa ser executada para cada nova amostra ou lote de amostras.
- **Escalonamento de dados:** O tempo para normalizar ou padronizar os dados antes de alimentá-los ao modelo. Assim como a engenharia de features, esta etapa é executada em tempo real para cada nova amostra.
- **Inferência do modelo:** O tempo que o modelo treinado leva para fazer uma previsão em uma nova amostra de dados pré-processada e escalonada. Essa é a métrica de latência mais direta e frequentemente a mais crítica para aplicações em tempo real.

O tempo total de detecção para uma amostra é a soma dos tempos citados acima. Em um cenário de borda, onde a tomada de decisão é contínua, o tempo médio por amostra para essas etapas é o que realmente importa.

3.4 Algoritmos de ML

O trabalho em questão visa realizar detecção e mitigação de ataques por meio de análise de anomalias. Assim, adotamos os algoritmos: Árvores de decisão, Multilayer Perceptron (MLP), Support Vector Machine (SVM), Florestas Aleatórias, Naive Bayes e AdaBoost. Todos os algoritmos adotados são capazes de analisar e detectar padrões complexos. Começando pelas Árvores de Decisão, que são mais simples, é possível rastrear uma falha específica seguindo os ramos da árvore gerada (IBM, 2024a). Por outro lado, o Naive Bayes é computacionalmente eficiente e fornece saídas probabilísticas, o que pode ser muito útil para detectar falhas em aprendizado supervisionado. Esse algoritmo pertence à família dos modelos generativos e é amplamente utilizado em tarefas de classificação de texto (IBM, 2024b). Enquanto isso, os algoritmos MLP e SVM são altamente poderosos e capturam padrões intrincados Bishop e Nasrabadi (2006). Já o algoritmo Florestas Aleatórias, combina múltiplas árvores de decisão para aumentar a precisão e reduzir o risco de *overfitting*, semelhante ao *AdaBoost* Bishop e Nasrabadi (2006). Entretanto, esse pode generalizar melhor e obter maior eficiência computacional em pequenos conjuntos de dados.

4 TRABALHOS RELACIONADOS

A Inteligência Artificial (IA) possui uma gama de metodologias e aplicações amplas que permitem que máquinas simulem habilidades cognitivas e capacidades de resolução de problemas do cérebro humano, por meio da utilização de algoritmos de ML e Deep Learning (DL), que são um subconjunto da IA com um conjunto de técnicas que facilitam o aprendizado sistemático em máquinas.

A Tabela 2 apresenta um compilado de trabalhos relacionados envolto nos temas ML, DL, computação de borda, ataques DDoS de camada de aplicação, sistemas de detecção e sistemas IoT.

Tabela 2 – Compilado de trabalhos relacionados.

Autor	**Baseados em ML**	**Ambientes de borda**	**Camada de Aplicação**	**Detecção**	**Ambientes IoT**
Almeghlef *et al.* (2023)	✓	×	×	✓	✓
Yungaicela-Naula *et al.* (2021)	✓	×	✓	✓	✓
Raubitzek e Neubauer (2021)	✓	×	×	✓	×
Solanki e Chaudhari (2023)	✓	×	✓	✓	✓
Liu *et al.* (2021)	✓	×	✓	✓	✓
Kshirsagar e Kumar (2022)	✓	×	✓	✓	✓
Valdovinos *et al.* (2021)	✓	×	×	✓	✓
Kumari e Jain (2024)	✓	×	×	✓	✓
Krishna *et al.* (2021)	✓	✓	✓	×	✓
Kumar e Singh (2024)	✓	✓	×	✓	✓
Belachew *et al.* (2025)	✓	✓	✓	✓	✓
Jia *et al.* (2020)	✓	✓	✓	✓	✓
Filho (2018)	×	×	✓	✓	×

Fonte: Próprio autor.

A seguir são apresentados os trabalhos relacionados subdivididos em três subtópicos, sendo eles: trabalhos baseados em ML e séries temporais, trabalhos voltados a ambientes de borda e trabalhos focados em ataques de camada de aplicação.

4.1 Trabalhos baseados em ML e séries temporais

Os autores, no trabalho Almeghlef *et al.* (2023), apresentam um modelo de ML capaz de detectar ataques DDoS no Constrained Application Protocol (CoAP) com uma precisão de 98%. Sabemos, contudo, que o CoAP quando acoplado ao Datagram Transport Layer Security (DTLS) - que é um protocolo relativamente pesado - para proteger contra esse tipo de ataque, tem suas limitações. Para obtenção dessa precisão, foram aplicados os algoritmos *Naive Bayes*, Support Vector Classifier (SVC) que é um tipo de classificador SVM, Florestas aleatórias e Árvores de decisão, obtendo o resultado apresentado com este último. Contudo, não utilizam nenhuma técnica inovadora em busca de otimizações e métodos mais assertivos, principalmente no que diz respeito à velocidade de detecção.

Já no trabalho elaborado pelos autores Yungaicela-Naula *et al.* (2021), foram implementados modelos de ML (K-Nearest-Neighbor (KNN), SVM, florestas aleatórias) e DL (MLP, Convolutional Neural Network (CNN), Gated Recurrent Units (GRU) e Long Short-Term Memory (LSTM)) com uma arquitetura Software-Defined Network (SDN) na camada de aplicação para detecção de DDoS, alcançando uma taxa de detecção de 95% obtida pelos modelos GRU e LSTM. Entretanto, embora o uso de SDN traga vantagens, essa arquitetura também é suscetível a ataques DDoS, considerando que existe um controlador centralizado, o qual já foi separado do Intrusion Detection System (IDS) para evitar sobrecarga. E ainda, os modelos de DL podem ser pesados para análises em tempo real em dispositivos de borda, que não foram investigados no trabalho.

No estudo feito por Raubitzek e Neubauer (2021), um modelo de rede neural artificial com Backpropagation Neural Network (BPNN) foi usado para um estudo de acoplamento de magnetosfera do vento solar e do expoente de Hölder para analisar as propriedades de escala local e singularidade, sendo este usado como uma *feature* de entrada da rede neural, auxiliando a melhorar o desempenho do algoritmo. Além disso, o expoente de Hölder pode identificar variações locais com mais facilidade, sendo uma técnica atrativa para ataques furtivos e com baixa taxa, conseguindo aferir alterações sutis no tráfego e identificar ataques. Outros métodos, como combinar Hölder com uma análise de entropia, podem aumentar ainda mais a precisão, mas com pouco custo computacional, viabilizando a utilização em dispositivos de borda (RAUBITZEK; NEUBAUER, 2021).

Vários autores realizaram trabalhos utilizando análise de entropia para detecção de ataques DoS e DDoS (SOLANKI; CHAUDHARI, 2023; LIU *et al.*, 2021; KSHIRSAGAR;

KUMAR, 2022; VALDOVINOS *et al.*, 2021). Os autores Solanki e Chaudhari (2023), porém, realizaram um estudo para detectar e mitigar ataques DDoS usando análise de entropia atrelada ao coeficiente de Hurst para tráfego em tempo real, baseando-se em autossimilaridade do tráfego. Os autores afirmam que o modelo aplicado obteve precisão de 99,75% e baixo atraso, sugerindo uma validação cruzada de técnicas de Hurst e entropia, mas testando no mesmo conjunto de dados, ou seja, o mesmo cenário gerador dos dados. Assim, o estudo não garante que o modelo treinado generalize bem para dados advindos de cenários não conhecidos, sendo uma característica relevante de tráfegos de rede que evoluem e se modificam bastante com o tempo, sendo suscetível a ataques *zero-day*. O artigo também ressalta alto desempenho computacional, entretanto, não especifica as configurações de hardware onde foram aferidas as métricas de desempenho, sendo informações relevantes para uma avaliação minuciosa. Além disso, não avalia a generalização dos dados em um conjunto de dados com cenário realmente distinto do cenário de treinamento. E ainda, a autossimilaridade estatística não é tão sensível a mudanças locais usando monofractais e precisa de um grande volume de dados históricos. Entretanto, usar o conceito de multifractais pode apresentar diferentes graus de irregularidades locais de um sinal, usando uma dimensão fractal em vez de um único valor. Um expoente que usa o conceito de multifractais é conhecido como expoente local de Hölder (YU; QI, 2011). Contudo, o expoente de Hölder trabalha com regularidade local, conseguindo identificar variações em séries temporais com poucos dados históricos e com nível de sensibilidade superior ao de Hurst. Além disso, o expoente de Hurst envolve múltiplas operações matemáticas complexas como transformadas de Fourrier e um grande volume de dados históricos e estatísticas de escala, tornando a análise pesada computacionalmente, dificultando implementações em tempo real em dispositivos de borda. Contudo, o expoente de Hölder não necessita de um grande número de amostras para estimar oscilações, calculando estimativas com pequenos segmentos de tráfego, reduzindo o custo computacional.

4.2 Trabalhos voltados a ambientes de borda

O trabalho realizado por Kumar e Singh (2024) apresenta um método com ML que afirma fornecer uma solução de segurança em IoT para servidores de borda. Contudo, não apresenta o hardware em que o experimento foi aplicado e as métricas de avaliação utilizadas no classificador não garantem que o trabalho de fato tenha conseguido desenvolver um bom classificador, uma análise mais aprofundada é necessária para identificar gargalos não apresentados

pelos autores.

Já o estudo apresentado por Belachew *et al.* (2025) propõe um classificador implementado em um servidor de borda em uma rede SDN e afirma um desempenho de 3,946 ms para detecção. Entretanto, utilizam um servidor de borda com elevadas capacidades computacionais para a implementação, destoando de propostas com bom custo-benefício. Além disso, não faz um *cross-dataseting* com as bases de dados abordadas no estudo.

Os autores Jia *et al.* (2020), no entanto, desenvolveram uma ferramenta de detecção de intrusões denominada *FlowGuard* combinando um *dataset* público e dados de tráfego gerados pelos autores para aumentar o conjunto de dados e buscar melhor precisão na identificação de ataques. Afirma ainda que os modelos são implementados em servidor de borda com capacidades computacionais superiores às de um computador pessoal, com 12 processadores, seis kernels e uma RAM instalada com 32,0 GB de memória disponível executando o sistema operacional Windows 10 de 64 bits. Cada processador está configurado com uma CPU Intel Core i7-8750H a 2,20 GHz, 2201 MHz. O servidor de borda usado no experimento, quando comparado a servidores de borda mais simples, como um Raspberry Pi, tem elevado poder computacional onerando demasiadamente o sistema e inviabilizando sistemas escaláveis onde seriam necessários vários dispositivos de borda atuando.

4.3 Trabalhos focados em ataques de camada de aplicação

Muitos estudos vêm sendo realizados buscando avaliar a aplicação de fractais e entropia em análise de tráfego de redes. O trabalho realizado por Filaretov e Chervova (2022), por exemplo, faz um estudo experimental sistematizado das propriedades fractais do tráfego legítimo e do tráfego típico para ataques DDoS de vários tipos, para esclarecer a eficácia das características fractais para detectar ataques cibernéticos. Este faz um comparativo de alguns métodos com fractais mostrando resultados positivos em análise de tráfego de redes em ataques como HTTP *Flood*, HTTP GET/POST e *SlowLoris*, mostrando que o estudo de fractais em tráfegos de redes, principalmente em ataques de camada de aplicação, pode ser efetivo. Entretanto, métodos usando o expoente local de Hölder não foram avaliados no trabalho. Contudo, essa técnica vem sendo investigada ao longo dos anos.

O autor Filho (2018) desenvolveu um método baseado em entropia de Shannon para detecção de ataques DDoS *Low Rate Slowloris*. Contudo, este aplica a técnica apenas analisando a distribuição de *IP's*, tendo mais eficácia em cenários com um grande número de atacantes.

Contudo, ataques de camada de aplicação como *Slowloris* e *SlowHTTPtest* podem conter uma quantidade de atacantes bem inferior a ataques massivos, dificultando ainda mais a detecção em cenários reais. Assim, extrair informações de outras *features* mais impactantes para a aplicação, como as *flags* que carregam uma grande quantidade de informação e são adequadas para a aplicação da análise de entropia, não foi avaliado no estudo. Além disso, o autor cria seu próprio *dataset* e não faz uma validação temporal e testes de generalização com outros conjuntos de dados para evidenciar a qualidade de desempenho e generalização do seu método.

Kumari e Jain (2024) apresenta um *overview* completo e abrangente, com a análise de mais de 170 artigos, focando exclusivamente na detecção e mitigação de ataques DDoS na camada de aplicação. Tal foco se deve à dificuldade em detectar estes ataques, uma vez que se assemelham bastante ao tráfego legítimo e visam exaurir recursos computacionais sem gerar picos anômalos de tráfego na camada de rede. Além disso, enfatiza a utilização de técnicas de ML na defesa contra esses ataques. Os autores também propõem uma taxonomia detalhada de técnicas de detecção e mitigação.

Já Moraes (2023) apresenta um sistema de detecção em IoT baseado no parâmetro de Hurst intitulado DDSHP. No trabalho, o autor aborda a utilização de Hurst em redes SDN e avalia o desempenho e tempo de detecção do sistema proposto. O estudo afirma que o sistema tem eficiência em pequenos sistemas IoT, contudo, um controlador SDN somado ao sistema de detecção pode ser computacionalmente pesado, tanto que o experimento em questão utiliza um servidor central para tal aplicação (sem informações sobre *hardware*), necessitando de testes mais precisos para avaliar se o sistema poderia ser implementado em dispositivos de borda. Além disso, o experimento 3 realizado nesse trabalho avalia o tempo de detecção que, para diferentes cenários, na maioria dos casos o sistema DDSHP leva menos de 1 minuto para detectar a ocorrência. Contudo, aumenta o tempo de detecção ao aumentar a complexidade da rede. E ainda, não discrimina o *hardware* onde foi avaliado o desempenho do sistema, inviabilizando um comparativo ou métrica para implementação real em dispositivos de borda que possuem capacidade de processamento inferior.

Diante do estudo realizado, serão abordadas várias técnicas temporais, tais como o expoente de Hölder, análise de entropia de Shannon, ARIMA e suavização de dados com MME, usando análise local de tráfego de rede para investigar e validar análises em tempo real de alto desempenho em dispositivos de borda.

5 METODOLOGIA

5.1 Visão da Abordagem Experimental

A metodologia experimental adotada na pesquisa visa incrementar múltiplos cenários de testes, permitindo uma avaliação mais aprofundada e completa dos modelos propostos. Além disso, o rigor científico quanto a uma validação cruzada, objetivando garantir que o melhor modelo foi usado, buscando a transferibilidade por meio de múltiplos *datasets* sem retreinamento e aplicabilidade prática em ambientes com recursos computacionais limitados. Os experimentos abordados incorporam combinações de técnicas de séries temporais, visando encontrar modelos que permeiem com melhor generalização de dados entre os *datasets* sem a necessidade de retreinamento.

5.2 Datasets e Preparação de Dados

Tendo em mãos toda a metodologia para realizar a pesquisa, resta aplicar as técnicas e algoritmos em uma base de dados, preferencialmente com dados reais. Partindo dessa premissa, vários *datasets* foram avaliados em alguns critérios, como:

- Atualidade dos dados;
- Ataques DoS e DDoS na camada de aplicação;
- Volume de dados;
- Qualidade dos dados;
- Padronização de *features* em *datasets* distintos para aplicar *cross-dataseting* entre os mesmos.

A Tabela 3 apresenta um *overview* de *datasets* analisados, visando encontrar bases de dados sólidas para aplicar na pesquisa. Apesar das limitações de cada *dataset*, todos são considerados aplicáveis. Entretanto, buscou-se *datasets* que possuam padrões semelhantes de *features*, por exemplo, para implementar e avaliar o desempenho dos modelos em *cross-dataseting*, observando e buscando uma otimização temporal e melhor generalização dos dados. Ao final, foram adotados 3 *datasets* por possuírem padronização de *features* e contextos de cenários distintos em cada base de dados. Para padronização de features, todos utilizam uma ferramenta para extrair os recursos intitulada *CICFlowMeter*, resumida em 80 *features* de análise de tráfego de rede.

Tabela 3 – Comparativo de Datasets para Detecção de Ataques.

Detalhamento de Datasets de Ataques								
Referência	**Dataset**	**Disponível**	**Real, sintético ou simulado.**	**Rótulo**	**Volumétricos**	**Baseados Protocolo**	**Aplicações**	**Limitações**
((CIC), 2017)	CIC-2017	✓	Real	✓	✓	✓	✓	Dataset com dados reais, com comportamento abstrato de 25 usuários, foco principal em ataques DoS e DDoS de forma geral, não no contexto IoT.
(Canadian Institute for Cybersecurity (CIC), 2018)	CIC-2018	✓	Real	✓	✓	✓	✓	O conjunto de dados final inclui sete cenários de ataque diferentes: força bruta, heartbleed, botnet, DoS, DDoS, ataques web e infiltração da rede interna. A infraestrutura de ataque inclui 50 máquinas e a organização vítima possui 5 departamentos, incluindo 420 máquinas e 30 servidores.
(NETO *et al.*, 2023)	CIC-2023	✓	Real	✓	✓	✓	✓	Dados coletados em ambiente controlado, não retratando uma rede real com precisão. As features são extraídas de pacotes e fluxos específicos e nem todas as métricas possíveis são abrangidas.
(DATAPORT, 2018)	Bot-IoT	✓	Real e Simulado	✓	✓	✓	✓	Desbalanceamento de dados, ausência de rotulagem dos dados em algumas instâncias.
(KANG *et al.*, 2019)	IoT-NIDS	✓	Real	✓	✓	✓	✓	Documentação do dataset insuficiente, dados oriundos de ambiente controlado, dados desbalanceados.
(GUERRA-MANZANARES *et al.*, 2020)	MedBIoT	✓	Simulado	✓	✓	✓	✓	Os dados são simulados, o *malware* usado para infectar os dispositivos é dependente da arquitetura ARM, limitando a aplicabilidade do dataset em arquiteturas diferentes.
(MOUSTAFA *et al.*, 2020)	IoT-23	✓	Real	✓	✓	✓	✓	O dataset parece ter uma estrutura abrangente para entendimento de comportamentos, mas apresenta limitações quanto à sua abrangência e detalhes sobre certos tipos de ataques DDoS.
(VACCARI *et al.*, 2020)	MQTT	✓	Simulado	✓	✓	✓	✓	O dataset é ausente de ataques zero-day, classes desbalanceadas e criado em ambiente simulado.
(FERRAG MOHAMED AMINE E FRIHA, 2022)	Edge-IIoTSet	✓	Simulado	✓	✓	✓	✓	O dataset é gerado por meio de ferramentas de testes de rede, realizado em ambiente controlado e possui dados desbalanceados.

Fonte: Próprio autor.

5.2.1 Cenário CIC-IDS-2017

O conjunto de dados CIC-IDS-2017 contém ataques comuns e tráfego benigno que se assemelham a dados do mundo real. Além disso, utiliza resultados de análise de tráfego com o CICFlowMeter - uma ferramenta de análise de tráfego, também usada na padronização das *features* - com fluxos rotulados com base no registro de data e hora, IPs de origem e destino, portas de origem e destino, protocolos e ataques. A topologia usada para coleta de dados é composta por uma rede completa que inclui modem, *firewall*, *switches*, roteadores e a presença de uma variedade de sistemas operacionais, como Windows, Ubuntu e Mac OS X. O tráfego completo tendo um agente de criação de perfil de usuário e 12 máquinas diferentes na rede de vitimas e ataques reais da rede de ataque. Para este *dataset* temos dois ataques de camada de aplicação, sendo eles os ataques DoS *Slowloris* e *SlowHTTPtest*. A Figura 7 apresenta a arquitetura do *testbed* usada para criar o *dataset*.

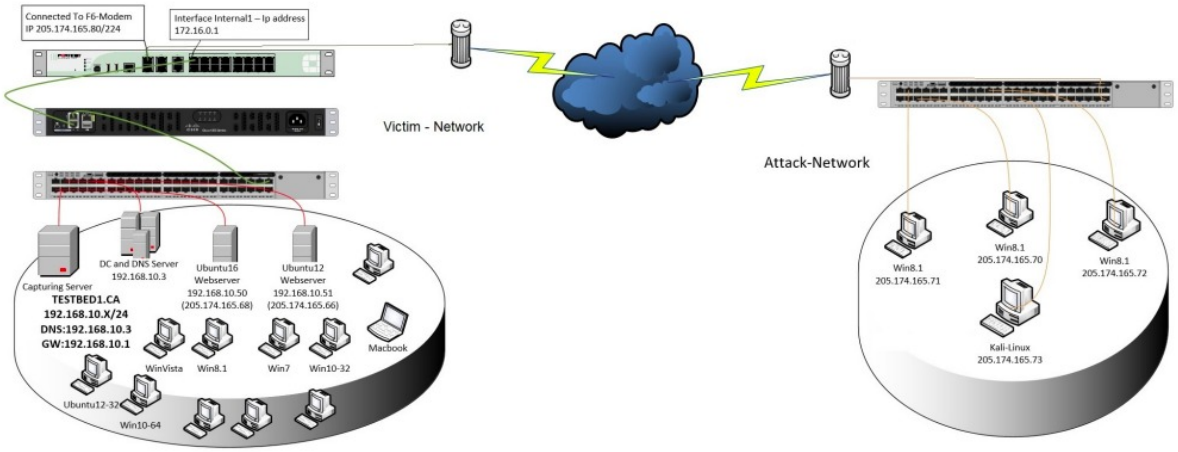

Figura 7 – Arquitetura do *testbed* (SHARAFALDIN *et al.*, 2018).

5.2.2 Cenário CSE-CIC-IDS2018

No conjunto de dados CSE-CIC-IDS2018, foram usados perfis para criar conjuntos de dados de forma sistemática, contendo descrições detalhadas de intrusões e modelos abstratos de distribuição para aplicações, protocolos ou entidades de rede de nível inferior. Para os ataques de negação de serviço, foram utilizadas *Slowloris* e LOIC como principais ferramentas que comprovadamente tornam servidores web completamente inacessíveis usando uma única máquina invasora. O *Slowloris* começa estabelecendo uma conexão TCP completa com o

servidor remoto. A ferramenta mantém a conexão aberta enviando requisições HTTP válidas e incompletas ao servidor em intervalos regulares para impedir o fechamento dos *sockets*. Como qualquer servidor web tem uma capacidade finita de atender conexões, será apenas uma questão de tempo até que todos os *sockets* sejam usados e nenhuma outra conexão possa ser estabelecida. Além disso, o HOIC é outra aplicação famosa que pode lançar ataques DoS contra sites. A Figura 8 a seguir mostra a arquitetura do cenário de ataque.

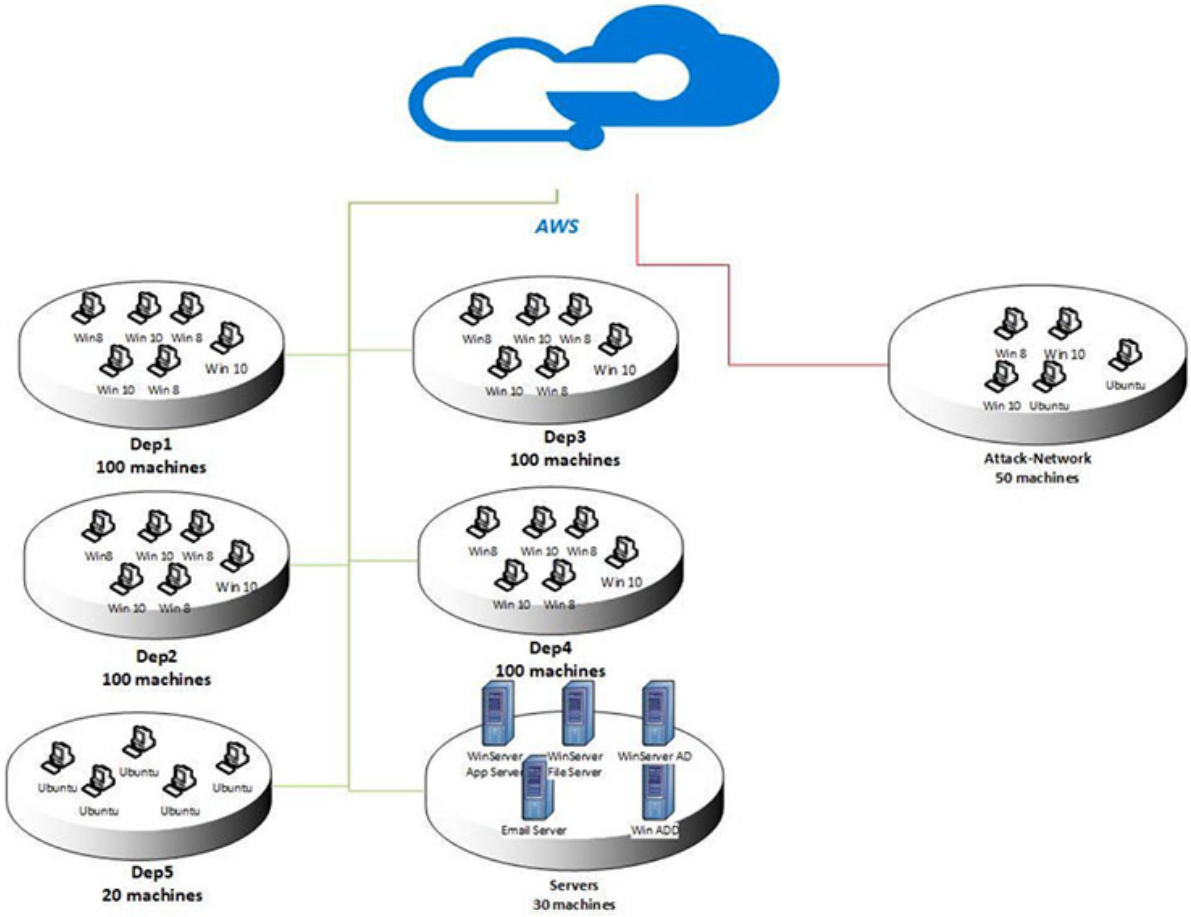

Figura 8 – Topologia da rede do CSE-CIC-IDS2018

5.2.3 Cenário CIC-IDS-2023

O CIC-IDS2023 é um conjunto de dados sobre ataques de IoT para fomentar o desenvolvimento de aplicações de análise de segurança em operações reais de IoT. Para isso, 33 ataques são executados em uma topologia de IoT composta por 105 dispositivos.

Este *dataset* Neto *et al.* (2023) se destaca em vários aspectos, são eles:

- Utilização de uma gama relevante de dispositivos reais para captura de dados (105 dispositivos).

- Coleta de dados realizada em ambiente real;
- Topologia de rede projetada para imitar redes IoT do mundo real, aumentando aplicabilidade dos modelos treinados;
- Diversidade de 33 ataques com 7 categorias principais;
- Execução dos ataques por dispositivos IoT contra dispositivos IoT;
- Abrange ataques em tempo real, possibilitando utilização em dispositivos de borda para detecção em tempo real;
- Presença de ataques DDoS nas categorias volumétricos, baseados em protocolo e na camada de aplicação.

A Figura 9 mostra a topologia usada para criar o *dataset* baseado em IoT.

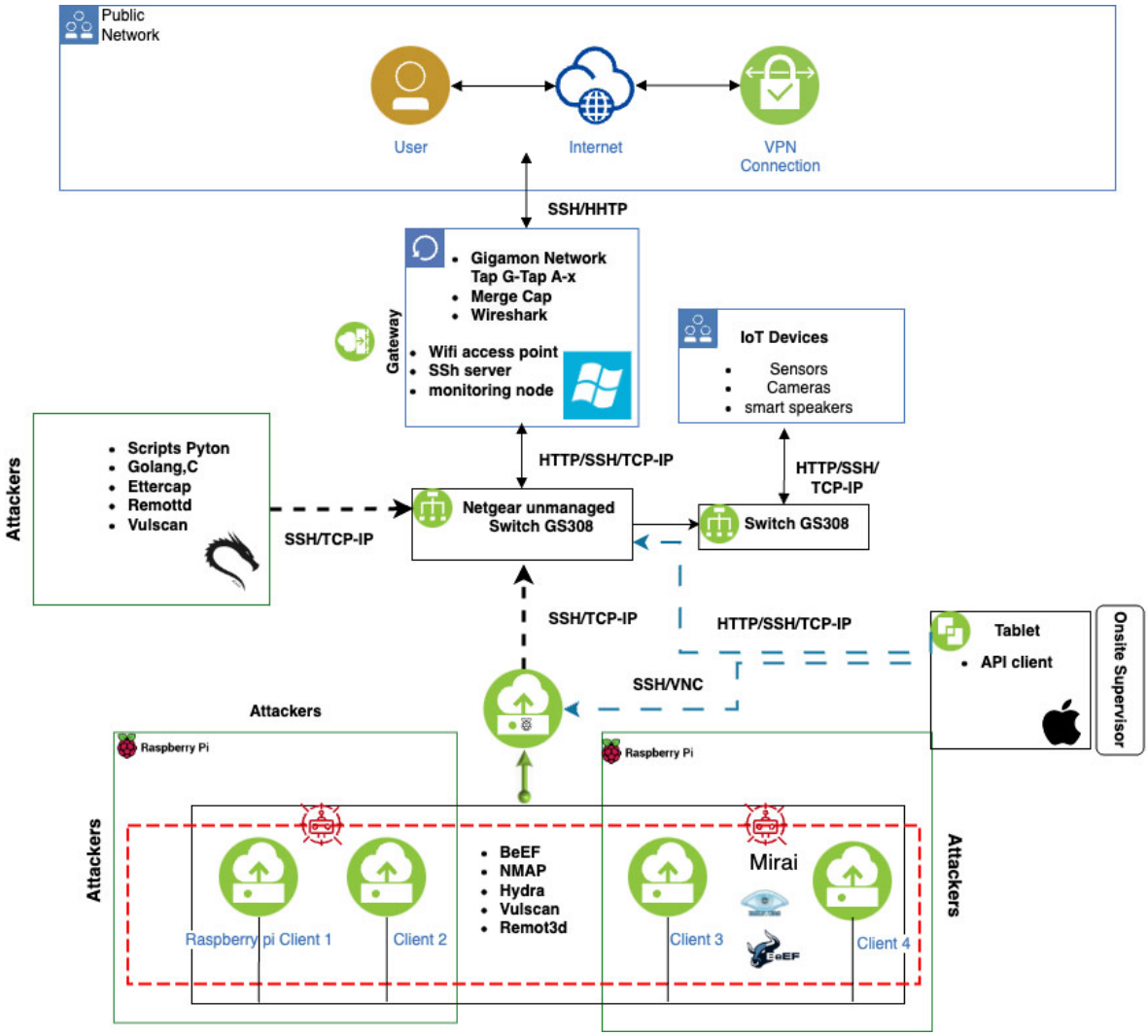

Figura 9 – Topologia da rede do CIC-2023

5.2.4 Configurações abordadas no experimento

O experimento em questão aborda três *datasets* de análise de tráfego de redes com ataques cibernéticos diversos, entre eles, os aplicados à pesquisa em questão. Os dados são padronizáveis, visando à aplicação de *cross-dataseting* entre os mesmos por meio de ferramenta de extração de recursos de tráfegos de rede, a CICFlowMeter. Além disso, foram abordadas algumas combinações de técnicas de séries temporais escolhidas cuidadosamente e implementadas sob alguns algoritmos de ML. A partir de então, iremos fazer menção a cada configuração no decorrer da metodologia e resultados, conforme lista a seguir:

1. **Configuração 1:** Dados originais do CICFlowMeter, médias móveis simples e exponenciais, análise de entropia de Shannon, ARIMA e expoente local de Hölder;
2. **Configuração 2:** Dados originais do CICFlowMeter, médias móveis simples e exponenciais, análise de entropia de Shannon e ARIMA;
3. **Configuração 3:** Dados originais do CICFlowMeter, médias móveis simples e exponenciais, análise de entropia de Shannon e expoente local de Hölder;
4. **Configuração 4:** Dados originais do CICFlowMeter, análise de entropia de Shannon e expoente local de Hölder;
5. **Configuração 5:** Dados originais do CICFlowMeter, médias móveis simples e exponenciais e análise de entropia de Shannon.

5.2.5 Pré-Processamento e Padronização de Datasets

Para realização deste estudo e implementação de *cross-dataseting* foi necessário realizar um apanhado de muitas bases de dados, buscando *datasets* com *features* compatíveis, possibilitando a aplicação de *cross-dataseting*. Assim, encontradas as bases de dados, foram tomadas algumas medidas básicas de padronização, embora as bases de dados escolhidas sejam oriundas do mesmo repositório e utilizem metodologia semelhante de extração de recursos, foi necessário realizar alguns ajustes para possibilitar a aplicação.

O procedimento apresentado na Figura 10 foi realizado em todos os *datasets* utilizados para o desenvolvimento da pesquisa, com a finalidade de padronizar os mesmos com relação às *features*. O CIC-2017 foi adotado como base de dados principal para treinamento, visando uma evolução temporal natural para generalização dos dados.

Após a etapa de padronização, uma nova etapa de extração de *features* foi realizada,

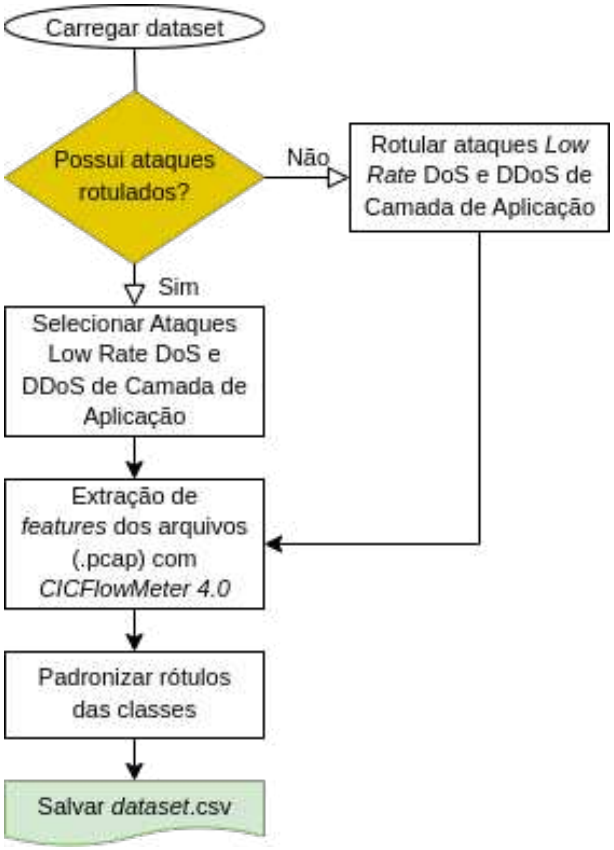

Figura 10 – Pré-processamento dos *datasets*.
Fonte: Próprio autor.

desta vez aplicando algumas técnicas de análise de séries temporais, mencionadas nas configurações apresentadas no subtópico 5.2.4. A metodologia adotada até os testes finais possui particularidades bem específicas devido às várias combinações realizadas visando uma melhor generalização temporal dos modelos de ML. Assim, seguindo os passos realizados na pesquisa, foi realizada a extração de *features* no (CIC) (2017), para o caso com a maior combinação de técnicas avaliado no experimento, sendo esta a Configuração 1.

Para otimizar o tempo e cruzar uma maior gama de possibilidades nas configurações, expandimos o *dataset* em mais de 11 vezes o seu tamanho original visando a redução de dimensionalidade específica em cada Configuração abordada. Após carregar o *dataset* padronizado com o procedimento da Figura 10, é realizado um pré-processamento de dados quanto ao balanceamento de classes, pois devido à natureza dos dados, é extremamente mais comum dados de tráfego normal a dados de tráfego de ataques. Após isso, houve um cuidado em separar as *features* puramente binárias (salvas em S1) para aplicações com entropia de Shannon e numéricas para aplicação das demais técnicas de análises temporais de forma correta. Assim, o pipeline de transformação de *features*, iniciando com ARIMA ocorreu primeiramente testando e selecionando as *features* estacionárias e salvando-as em S2. Posteriormente, encontrando os

melhores parâmetros (d, p, q) para posterior aplicação e salvando em S3. Além disso, foram selecionadas as 5 *features* numéricas mais relevantes para aplicação do expoente local de Hölder e salvas em S4. Seguindo o fluxo, as *features* a partir de MME, Hölder, ARIMA e entropia foram criadas. Após isso, foi realizada a união das *features* criando um novo *dataset* robusto e completo, salvo em S5.

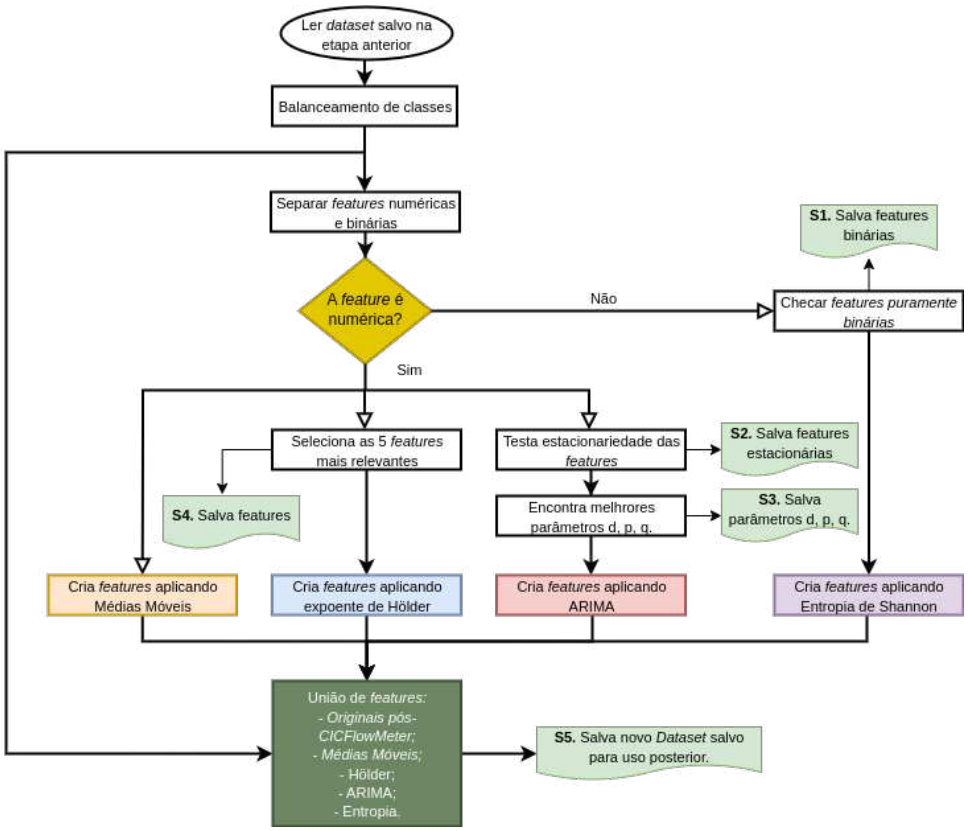

Figura 11 – Pré-processamento inicial dos *datasets*.
Fonte: Próprio autor.

Importante ressaltar que na transformação de *features*, usamos várias janelas distintas em cada técnica aplicada individualmente visando encontrar, ao final, a melhor combinação de *features* para treinar os algoritmos, inclusive médias móveis nas *features* binárias. Os valores das janelas foram escolhidos após diversos testes individuais de desempenho e, para testes mais robustos, usamos todas as possibilidades em uma base de dados maior, com objetivo de reduzi-la para implementações com melhor desempenho. As janelas escolhidas entre outras características podem ser vistas na Tabela 4. Após as etapas executadas no fluxograma apresentado na Figura 11, a etapa de pré-processamento de cada configuração foi realizada conforme mostra a Figura 12 para todas as configurações, com a particularidade de que a Configuração 1 usa todas as técnicas sugeridas e, portanto, não dropa nenhum tipo antes de selecionar as que possuem maior correlação.

Tabela 4 – Técnicas de Engenharia de *Features* e Janelas Utilizadas

Técnica	**Tamanhos das Janelas**
Médias Móveis	3, 6, 9 e 12
ARIMA	1, 2, 3, 4 e 5
Entropia de Shannon	10, 20, 30, 40 e 50
Hölder	5

Fonte: Próprio autor.

As *features* a serem dropadas inicialmente neste processo seguem as técnicas abordadas em cada configuração conforme item 5.2.4.

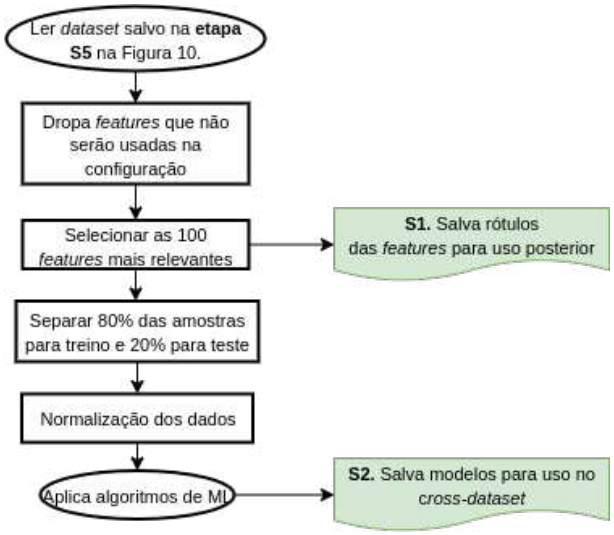

Figura 12 – Pré-processamento das configurações 1 a 5.
Fonte: Próprio autor.

Os procedimentos realizados nos fluxogramas da Figura 11 e da Figura 12 foram aplicados no *dataset* CIC-2017. O próximo subtópico abordará a metodologia utilizada na preparação dos dados para o *cross-dataseting*.

5.3 Protocolo de Transferibilidade (*Cross-datasetting*)

É de suma importância o manejo correto dos dados para a realização do *cross-dataseting* para garantir a qualidade dos resultados. Diante disso, a Figura 11 apresenta o fluxograma com a metodologia que deve ser aplicada também ao *dataset* CIC-2018 inicialmente, a fim de obter um novo *dataset* robusto nesta base de dados.

Todas as configurações para o *cross-dataseting* devem seguir o disposto na metodologia da Figura 13 para cada configuração disposta no item 5.2.4.

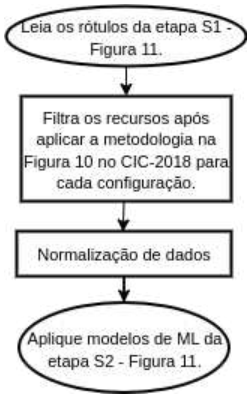

Figura 13 – Pré-processamento do segundo *dataset* para o *cross-dataseting*.
Fonte: Próprio autor.

5.4 União de *dataset's*

Após encontrados os modelos atemporais, obtendo alto desempenho no *cross-datasetting*, é realizada uma união dos *datasets* CIC-2017, CIC-2018 e CIC-2023, contendo nos dois primeiros ataques DoS Slowloris e SlowHTTPtest e no CIC-2023 ataques DDoS Slowloris. Após esta junção, realizamos a mesma metodologia de treinamento inicial do CIC2017, mas agora com este novo *dataset*, propondo então um modelo de ML robusto e atemporal. Para realizar a união das bases de dados, utilizamos a base de dados gerada, para todos os casos, na **etapa S5** da Figura 11. Após isso, seguimos os mesmos passos apresentados na Figura 12 para a Configuração 5, que obteve melhor desempenho.

5.5 A Ferramenta CICFlowMeter

A Tabela 5 mostra todas as *features* extraídas do CICFlowMeter em todos os *datasets* utilizados, mostrando as quantidades, categorias das mesmas após a aplicação da ferramenta e uma breve descrição sobre grupo. As novas *features* extraídas a partir destas irão levar consigo o nome inicial e adicionar as técnicas de séries temporais e os tamanhos das janelas utilizadas na sua transformação, conforme a Tabela 4.

5.6 Configurações Experimentais

Para cada configuração aplicada, conforme as técnicas já mencionadas na Tabela 4 um conjunto de 100 *features* foi selecionado aplicando a técnica f_classif do *Scikit-Learn*. Na Tabela 6 é possível analisar as *features* geradas e selecionadas em cada técnica, janela e configuração.

Tabela 5 – Categorização das Features Extraídas pelo CICFlowMeter no Dataset CICIDS2017

Qtde	Categoria de feature	Descrição
3	Features Básicas de Fluxo	Incluem features que descrevem características fundamentais do fluxo como Flow duration (duração em microssegundos), Flow Byte/s (bytes por segundo) e Flow Packets/s (pacotes por segundo).
4	Features de Contagem de Pacotes	Incluem features que descrevem quantidades totais como total Fwd Packet, total Bwd packets, total Length of Fwd Packet e total Length of Bwd Packet para análise de volume de tráfego.
8	Features Estatísticas de Tamanho	Incluem features que descrevem estatísticas de tamanho de pacotes nas direções forward e backward, abrangendo Fwd/Bwd Packet Length Min, Max, Mean e Std para análise de distribuição de tamanhos.
14	Features de Tempo Inter-Arrival	Incluem features que capturam padrões temporais como Flow IAT Mean/Std/Max/Min e Fwd/Bwd IAT com estatísticas completas (Min, Max, Mean, Std, Total) para análise de comportamento temporal.
4	Features de Flags TCP Direcionais	Incluem features que contam flags específicas do protocolo TCP nas direções forward e backward, como Fwd/Bwd PSH flag e Fwd/Bwd URG Flag para análise de comportamento de protocolo.
4	Features de Cabeçalho e Taxa	Incluem features relacionadas aos cabeçalhos como Fwd/Bwd Header Length e taxas de pacotes por segundo (FWD/Bwd Packets/s) para análise de overhead e performance.
5	Features de Características Gerais	Incluem features que descrevem características gerais dos pacotes como Min/Max Packet Length, Packet Length Mean, Std e Variance para análise estatística geral.
8	Features de Contadores de Flags	Incluem features que contam diferentes tipos de flags TCP no fluxo completo como FIN, SYN, RST, PSH, ACK, URG, CWR e ECE Flag Count para análise detalhada do protocolo.
4	Features de Análise de Segmentos	Incluem features que analisam segmentos e transferências como down/Up Ratio, Average Packet Size e Avg Fwd/Bwd Segment Size para análise de padrões de comunicação.
6	Features de Transferência Bulk	Incluem features que analisam transferências em massa como Fwd/Bwd Avg Bytes/Bulk, AVG Packet/Bulk e AVG Bulk Rate para detecção de transferências de grande volume.
4	Features de Sub-fluxos	Incluem features que analisam sub-fluxos como Subflow Fwd/Bwd Packets e Subflow Fwd/Bwd Bytes para análise de comportamento interno do fluxo.
4	Features de Janela TCP	Incluem features específicas do TCP como Init Win bytes forward/backward, Act data pkt forward e min seg size forward para análise de controle de fluxo TCP.
8	*Features* de Atividade/Inatividade	Incluem *features* que capturam padrões temporais de atividade como Active/Idle Min, Mean, Max e Std para análise de comportamento de sessão e detecção de anomalias.
76	**Total de *Features***	**O CICFlowMeter extrai 76 *features* estatísticas organizadas em 13 categorias principais, fornecendo análise completa e detalhada do comportamento de fluxos de rede para sistemas de detecção de intrusão.**

Fonte: Próprio autor.

Tabela 6 – Descrição e Composição dos Conjuntos de *Features* por Configuração.

Configuração	Descrição do Conjunto
Configuração 1	*Features* Médias Móveis janela 3 (4), *Features* Médias Móveis janela 6 (6), *Features* Médias Móveis janela 9 (7), *Features* Médias Móveis janela 12 (15), *Features* com entropia janela 10 (2), *Features* com entropia janela 20 (2), *Features* com entropia janela 30 (2), *Features* com entropia janela 40 (2), *Features* com entropia janela 50 (2), *Features* com ARIMA *forecast* 1 (9), *Features* com ARIMA *forecast* 2 (9), *Features* com ARIMA *forecast* 3 (9), *Features* com ARIMA *forecast* 4 (9), *Features* com ARIMA *forecast* 5 (9), *Features* com ARIMA *fitted* 1 (9), *Features* com Hölder janela 5 (2), *Features* normais do CICFlowMeter (2).
Configuração 2	*Features* Médias Móveis janela 3 (4), *Features* Médias Móveis janela 6 (6), *Features* Médias Móveis janela 9 (8), *Features* Médias Móveis janela 12 (16), *Features* com entropia janela 10 (2), *Features* com entropia janela 20 (2), *Features* com entropia janela 30 (2), *Features* com entropia janela 40 (2), *Features* com entropia janela 50 (2), *Features* com ARIMA *forecast* 1 (9), *Features* com ARIMA *forecast* 2 (9), *Features* com ARIMA *forecast* 3 (9), *Features* com ARIMA *forecast* 4 (9), *Features* com ARIMA *forecast* 5 (9), *Features* com ARIMA *fitted* 1 (9), *Features* normais do CICFlowMeter (2).
Configuração 3	*Features* Médias Móveis janela 3 (12), *Features* Médias Móveis janela 6 (22), *Features* Médias Móveis janela 9 (25), *Features* Médias Móveis janela 12 (26), *Features* com entropia janela 30 (2), *Features* com entropia janela 40 (2), *Features* com entropia janela 50 (2), *Features* com Hölder janela 5 (2), *Features* normais do CICFlowMeter (3).
Configuração 4	*Features* com entropia janela 30 (2), *Features* com entropia janela 40 (2), *Features* com entropia janela 50 (2), *Features* com Hölder janela 5 (3), *Features* normais do CICFlowMeter (77).
Configuração 5	*Features* Médias Móveis janela 3 (13), *Features* Médias Móveis janela 6 (23), *Features* Médias Móveis janela 9 (25), *Features* Médias Móveis janela 12 (26), *Features* com entropia janela 10 (2), *Features* com entropia janela 20 (2), *Features* com entropia janela 30 (2), *Features* com entropia janela 40 (2), *Features* com entropia janela 50 (2) e *Features* normais do CICFlowMeter (3).

Fonte: Próprio autor.

Antes de qualquer coisa, é importante apresentar as *features* selecionadas após a implementação das técnicas de séries temporais realizadas em cada configuração, assim como os hiperparâmetros que proporcionaram melhor desempenho nos modelos para todos os algoritmos. Importante ressaltar que, uma vez encontrados os hiperparâmetros, foram padronizados para todas as outras aplicações, buscando um comparativo rigoroso de desempenho dos modelos diante das configurações aplicadas. A Tabela 7 mostra os hiperparâmetros utilizados em todos os

algoritmos aplicados.

Tabela 7 – Algoritmos e Hiperparâmetros Utilizados nos Experimentos

Algoritmo	**Hiperparâmetros**
MLP	'activation: 'tanh', 'alpha': 0.0001, 'batch_size': 'auto', 'beta_1': 0.9, 'beta_2': 0.999, 'early_stopping': True, 'epsilon': 1e-08, 'hidden_layer_sizes': (256,), 'learning_rate': 'constant', 'learning_rate_init': 0.05, 'max_fun': 15000, 'max_iter': 1000, 'momentum': 0.9, 'n_iter_no_change': 10, 'nesterovs_momentum': True, 'power_t': 0.5, 'shuffle': True, 'solver': 'adam', 'tol': 0.0001, 'validation_fraction': 0.1, 'verbose': True, 'warm_start': False
SVM	'C': 1.0, 'break_ties': False, 'cache_size': 200, 'coef0': 0.0, 'decision_function_shape': 'ovr', 'degree': 3, 'gamma': 'scale', 'kernel': 'rbf', 'max_iter': -1, 'probability': True, 'random_state': 42, 'shrinking': True, 'tol': 0.001, 'verbose': True
Naive Bayes	'var_smoothing': 1e-09
Árvores de Decisão	'criterion': 'entropy', 'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'
Florestas Aleatórias	'bootstrap': True, 'criterion': 'entropy', 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 100, 'oob_score': False, 'random_state': 42
AdaBoost	'algorithm': 'deprecated', 'learning_rate': 1, 'n_estimators': 50, 'random_state': 42

Fonte: Próprio autor.

6 RESULTADOS

Neste capítulo, é realizada uma análise quantitativa e qualitativa dos resultados. Na análise quantitativa, segue apresentando as métricas divididas em dois grupos, A e B. No Grupo A, estão os resultados de todas as configurações mencionadas no item 5.2.4 para o *dataset* CIC-2017. Já no Grupo B, estão os resultados de todas as configurações descritas no item 5.2.4 para o *dataset* CIC-2018. Ao final do capítulo, são abordadas detalhadamente as contribuições relativas ao trabalho dentro das temáticas de segurança em IoT, ML e computação de borda, tanto no que se refere à abordagem teórica quanto às implementações práticas do experimento.

6.1 Análise Quantitativa e Qualitativa

Esta seção apresenta as métricas de precisão, acurácia, sensibilidade, F1-Score, inferência dos modelos e desempenho computacional para todos os algoritmos, em todas as configurações, divididas em dois grupos (A e B). Avalia-se a eficácia de detecção, estabilidade dos modelos e o desempenho computacional em dispositivo de borda. Buscando uma visualização limpa dos resultados, devido a quantidade, é apresentado um gráfico com mapa de calor em formato de tabela, contendo um resumo de alguns resultados de todas as configurações para todos os algoritmos implementados, com as seguintes métricas: Acurácia, precisão, sensibilidade, F1-Score e inferência do modelo (desempenho computacional dos modelos aferidos individualmente no *Google Colab* em ms).

6.1.1 Configurações aplicadas ao CIC-2017 (Grupo A)

Aqui serão apresentadas as métricas relativas ao treinamento e teste utilizando o *dataset* CIC-2017. A partir da Figura 14, é possível observar que para todas as 5 configurações avaliadas nesse trabalho, tivemos um excelente desempenho inicial em quase todos os algoritmos implementados, tendo um desempenho abaixo do esperado apenas no algoritmo Naive Bayes para todas as configurações, porém, mais impactante na Configuração 4 sem a utilização das *features* geradas com médias móveis e ARIMA. Importante citar que nessa mesma figura, os dados de inferência dos modelos são referentes a plataforma Google Colab, sendo aplicados à tabela apenas como diferenciação dos modelos com melhor desempenho computacional de forma geral.

Os resultados iniciais demonstram que, no contexto do cenário da base de dados,

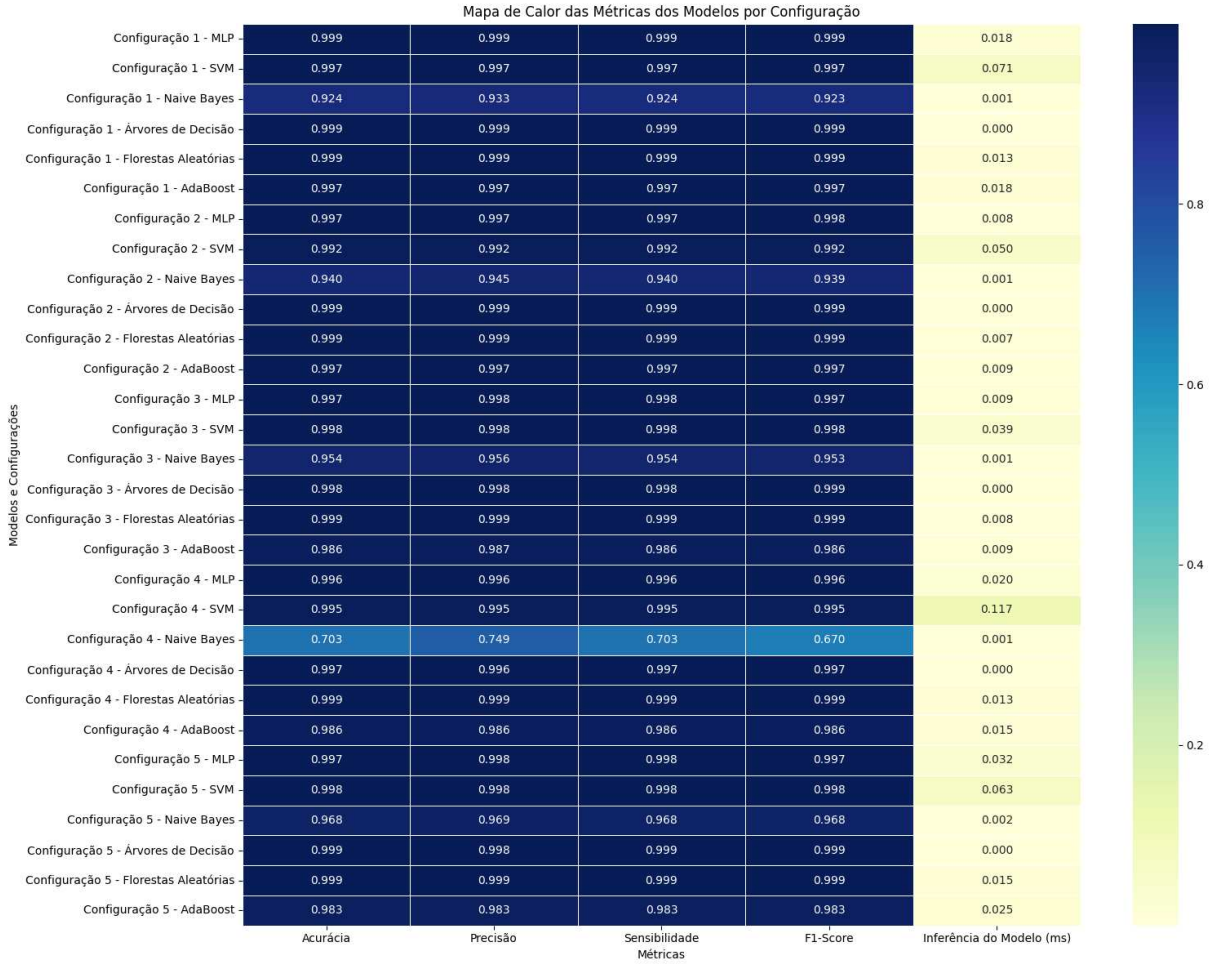

Figura 14 – Avaliação de desempenho com as métricas de acurácia, precisão, sensibilidade, F1-Score e inferência dos modelos de ML.

Fonte: Próprio autor.

todas as configurações obtiveram resultados excelentes, com métricas acima de 99% para precisão, acurácia, sensibilidade, F1-Score e intervalo de confiança na maior parte dos modelos, diferenciando-se apenas no desempenho computacional de cada pipeline em cada configuração.

6.1.2 Configurações aplicadas ao CIC2018 - Grupo B

Como mencionado anteriormente, neste subtópico serão apresentados os resultados das métricas refentes ao *cross-dataseting* com o CIC-2018, utilizando os modelos dos algoritmos treinados com o CIC-2017. Em uma rápida visualização, é perceptível a diferença de desempenho da maior parte dos algoritmos quando aplicados a dados fora do cenário de treinamento.

Entretanto, o desafio aqui é buscar uma modelagem que auxilie na detecção de novos ataques, ou seja, fora do contexto de dados do primeiro *dataset*. Mesmo com o desafio de um *cross-dataseting*, ainda foram encontrados modelos que tiveram alto desempenho, como pode

ser observado nas métricas mostradas no gráfico da Figura 15, com destaque para os modelos da Configuração 5, exceto o Naive Bayes.

Além disso, modelos isolados de algumas configurações, como MLP na Configuração 2 e AdaBoost na Configuração 4, também apresentaram alto desempenho dentro das condições avaliadas, evidenciando que os modelos com ARIMA e Hölder, combinados à entropia de Shannon, conseguem generalizar bem os dados para dados de tráfego de rede.

No entanto, o modelo ARIMA demanda mais poder computacional do que os demais modelos aplicados, como pode ser observado nos tempos de inferência dos modelos apresentados no gráfico apresentado na Figura 15, sendo Hölder uma melhor escolha entre os dois.

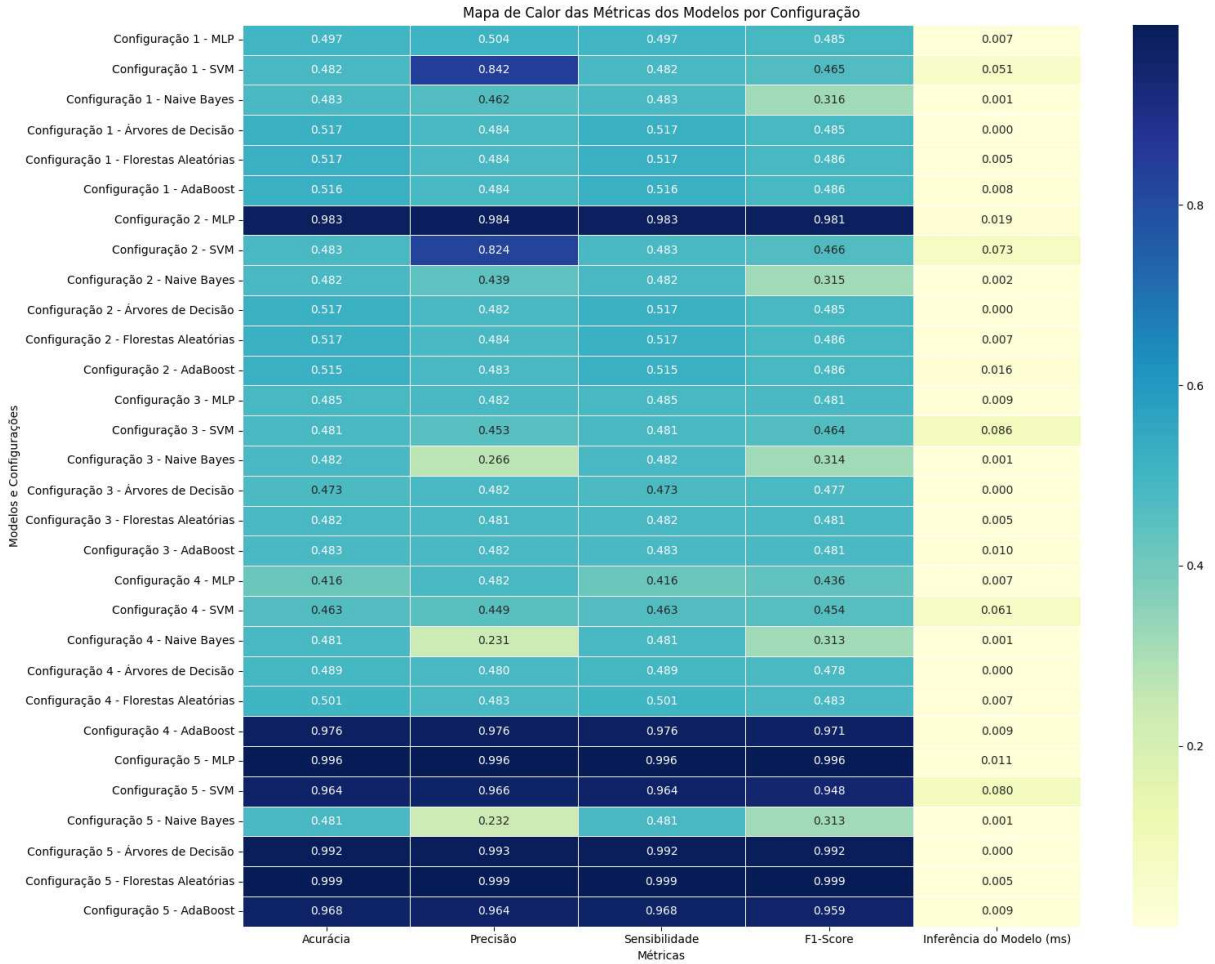

Figura 15 – Avaliação de desempenho com as métricas de acurácia, precisão, sensibilidade, F1-Score e inferência dos modelos de ML.

Fonte: Próprio autor.

A Configuração 5, por sua vez, se destaca entre as demais nesse desafio, apresentando desempenho inferior apenas em um dos seis algoritmos implementados, mostrando que a combinação de médias móveis e entropia de Shannon, com a metodologia aplicada, conseguiu

generalizar melhor os dados de tráfego de rede quando combinados.

Vale ressaltar que a metodologia conseguiu ainda generalizar cenários distintos, sendo uma alternativa interessante para avanços na detecção de ataques DoS e DDoS de camada de aplicação para sistemas IoT na borda.

Os valores de inferência dos modelos da Figura 14 e da Figura 15 só mostram qual modelo, já treinado, possui melhor desempenho de forma geral. Entretanto, a proposta deste trabalho é desenvolver uma plataforma para detecção de ataques DoS e DDoS na borda. Assim, o próximo tópico apresenta o desempenho computacional de todos os modelos, em todas as configurações, quando aplicados a um dispositivo de borda simples.

6.1.3 Análise do Desempenho Computacional na Borda

Os modelos apresentados, já treinados, foram implementados dentro de um dispositivo de borda simples, com capacidades computacionais bem inferiores às de outros existentes no mercado. O dispositivo escolhido, contudo, foi um Raspberry PI 3b+, que inclui um processador *Broadcom* BCM2837B0 quad-core Cortex-A53 de 64 bits com *clock* de 1.4 GHz, 1 GB de RAM LPDDR2, Wi-Fi dual-band (2.4GHz e 5GHz). No entanto, existem modelos da mesma linha com capacidades computacionais superiores, além de opções ainda mais robustas de dispositivos de borda voltadas ao desenvolvimento de IA. A Figura 16 apresenta os tempos, em milissegundos, do desempenho médio de detecção de cada modelo em todas as configurações abordadas, de acordo com cada pipeline.

Contudo, em um cenário real, haveria ainda a coleta de tráfego de rede e a aplicação do CICFlowMeter na extração de recursos, que não foram contabilizadas no experimento. Observe que o desempenho computacional, de forma geral, foi excelente, tendo em vista que se trata de um ataque lento, de difícil detecção e que está sendo aferido por meio de um dispositivo de borda com capacidades computacionais reduzidas e em tempo real. O pior caso não chega a 98 ms, enquanto o melhor desempenho ficou próximo de 7 ms. Ressalta-se, contudo, que, ao aplicar os algoritmos no dispositivo de borda, foi realizado o procedimento de extração de *features* apenas para as *features* selecionadas, considerando a **etapa S1** da Figura 12, uma vez que estas *features* são as mais relevantes em cada configuração. Dessa forma, foi possível obter um melhor desempenho computacional na borda.

As métricas apresentadas nas Figuras 14, 15 e 16 auxiliam na análise de desempenho de forma geral, mas por si só, não garantem que um modelo realmente tenha desempenho

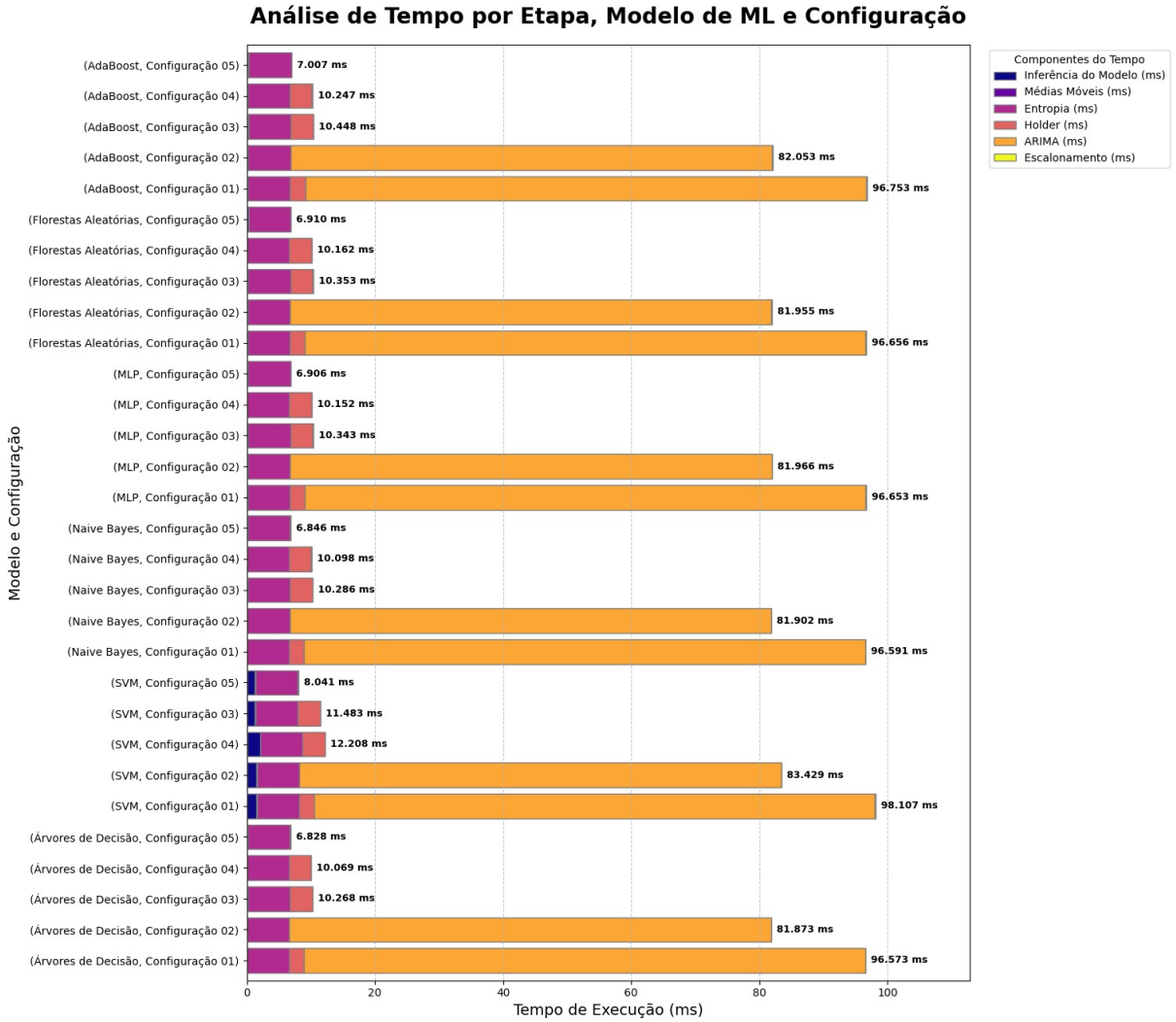

Figura 16 – Avaliação de desempenho computacional dos modelos implementados em todas as configurações em um Raspberry PI 3b+.

Fonte: Próprio autor.

satisfatório em um cenário real, sob uma perspectiva mais ampla. Contudo, direciona aos modelos de melhor desempenho, que neste caso, são especialmente os modelos da Configuração 5 em sua maior parte. Dessa forma, é possível avaliar detalhadamente os modelos de destaque, evidenciando aqueles que obtiverem alto desempenho e conseguiram generalizar de forma satisfatória os dados, principalmente diante do desafio aplicado. Assim, o próximo subtópico apresenta detalhadamente os resultados referentes à Configuração 5 para o caso B.

6.1.4 Análise Detalhada dos Modelos da Configuração 5 - Grupo B

Após a análise dos resultados obtidos, o estudo foi direcionado para uma avaliação mais aprofundada dos resultados da Configuração 5. Desta forma, a Figura 17 apresenta a matriz de confusão de todos os modelos aplicados nessa configuração do experimento.

Em uma avaliação minuciosa, observamos que, embora as métricas já apresentadas mostrem boa generalização, nem todos os modelos, de fato, teriam êxito em uma aplicação real. Veja, por exemplo, o modelo com MLP, que apresentou um quantitativo elevado de verdadeiros positivos e verdadeiros negativos. Contudo, registrou uma pequena quantidade de falsos negativos mais acentuada , especialmente em situações em que deveria classificar como DoS SlowHTTPTest, mas acabou classificando como DoS Slowloris. Entretanto, em uma classificação binária de ataque ou tráfego normal, esse pequeno montante de falsos negativos se torna menos relevante que o percentual apresentado, que possui um quantitativo total extremamente inferior. Isso demonstra que o modelo em questão possui potencial para aplicações reais. Apesar do desempenho notável do modelo MLP, outro modelo se destacou entre os demais: o modelo com o algoritmo Florestas Aleatórias. Observa-se no gráfico que seu desempenho no *cross-dataseting* obteve desempenho excepcional.

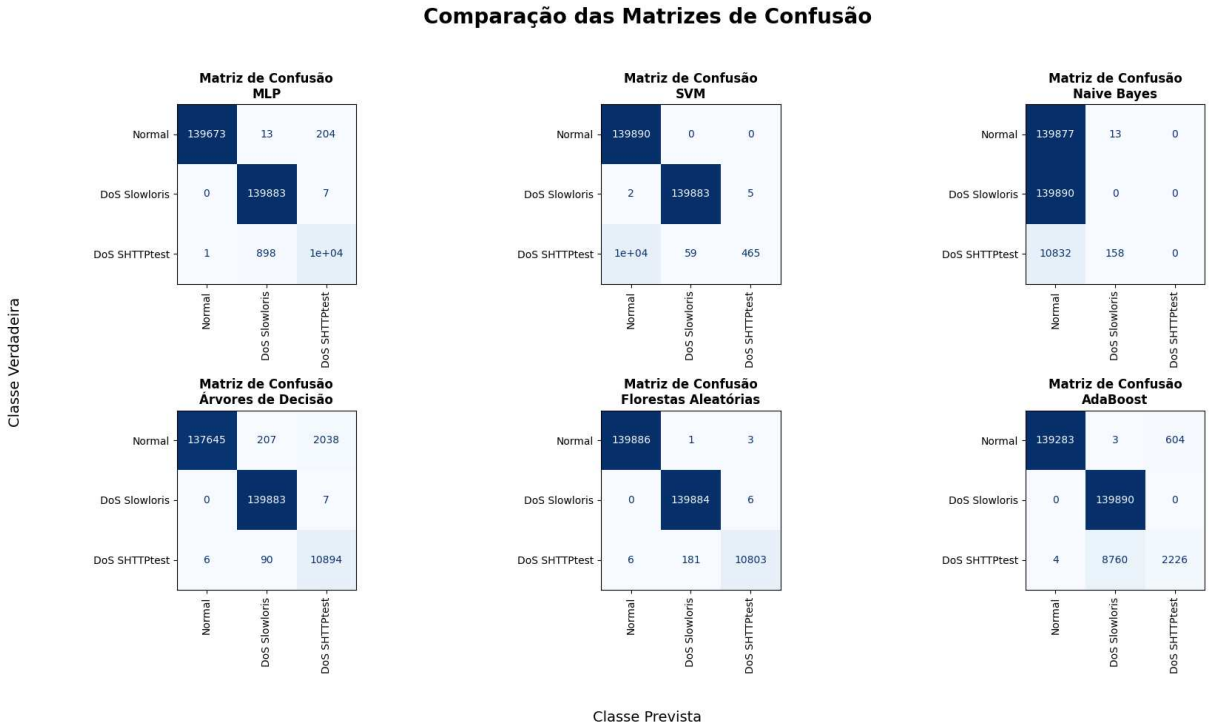

Figura 17 – Matriz de confusão dos modelos aplicados a Configuração 5.
Fonte: Próprio autor.

Observa-se que, para o algoritmo Florestas Aleatórias, os falsos positivos são praticamente zero em relação à amostra de dados, em que em uma situação real um tráfego legítimo seria classificado como ataque. Assim, evitaria o bloqueio de um possível cliente real do sistema ou o impedimento de alguma transação. Em relação aos falsos negativos, temos um ataque real que não foi classificado como ataque ou de um ataque por outro. Contudo, veja que os valores

também são próximos de zero, tornando-os irrelevantes em relação à quantidade da amostra de dados. Percebe-se também que o *recall* nesse ponto reduziu um pouco, não em relação a ataques não classificados, mas sim na classificação de um ataque por outro. Ainda assim, o percentual é mínimo, com uma taxa de FN de 1,3%, o que torna esse modelo extremamente confiável e garante que a maioria das ameaças reais sejam detectadas e neutralizadas nessa validação cruzada entre bases de dados diferentes.

Diante do exposto, para validar por completo os modelos com desempenho superior, a Figura 18 apresenta a curva ROC dos algoritmos aplicados à Configuração 5. Observa-se que o *trade-off* entre a taxa de verdadeiros positivos e a taxa de falsos positivos é excelente para os modelos em destaque (MLP e Florestas Aleatórias), o que torna a validação desses modelos em aplicações no mundo real totalmente possível.

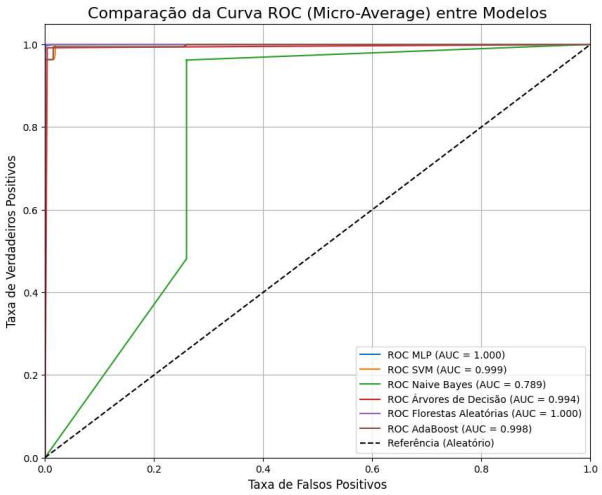

Figura 18 – Curva ROC dos modelos aplicados na Configuração 5.
Fonte: Próprio autor.

Um dos resultados significativos deste trabalho está no alto desempenho durante o *cross-dataseting* entre o CIC-2017 e o CIC-2018. Os resultados vão além de uma simples validação de performance, demonstrando uma forte capacidade de generalização do modelo. Essa robustez é um passo fundamental para viabilizar implementações da plataforma no mundo real, onde os perfis de tráfego estão em constante evolução.

A partir dos resultados obtidos e da análise criteriosa dos modelos em uma validação temporal, buscando melhorar e abranger uma gama maior de detecção de ataques de camada de aplicação, propõe-se a junção dos datasets CIC-2017, CIC-2018 e CIC-2023, com o objetivo de ampliar a capacidade de detecção de ataques na camada de aplicação. A proposta vai além da otimização dos modelos e da ampliação da cobertura de ataques: busca tornar o modelo

proposto mais robusto quanto à validação temporal e à generalização dos dados em cenários desconhecidos buscando mais que a otimização de modelos e abrangência de ataques, mas tornar o modelo proposto mais robusto quanto à validação temporal e generalização dos dados em cenários desconhecidos.

No tópico seguinte, serão apresentados os resultados dessa fusão dos *datasets*.

6.1.5 Modelo Robusto com dataset híbrido (CIC-2017, CIC-2018 e CIC-2023) - Grupo C

Agora que já foi identificada uma configuração que se destacou em relação às demais no *cross-dataseting*, foi realizada a fusão dos *datasets* para retreinar os modelos e melhorar a validação temporal dos mesmos, aumentando também a abrangência de ataques. A seguir, apresentamos na Figura 19 as métricas de acurácia, precisão, sensibilidade, F1-Score e o intervalo de confiança calculado em 10 *folds* de *cross-validation*.

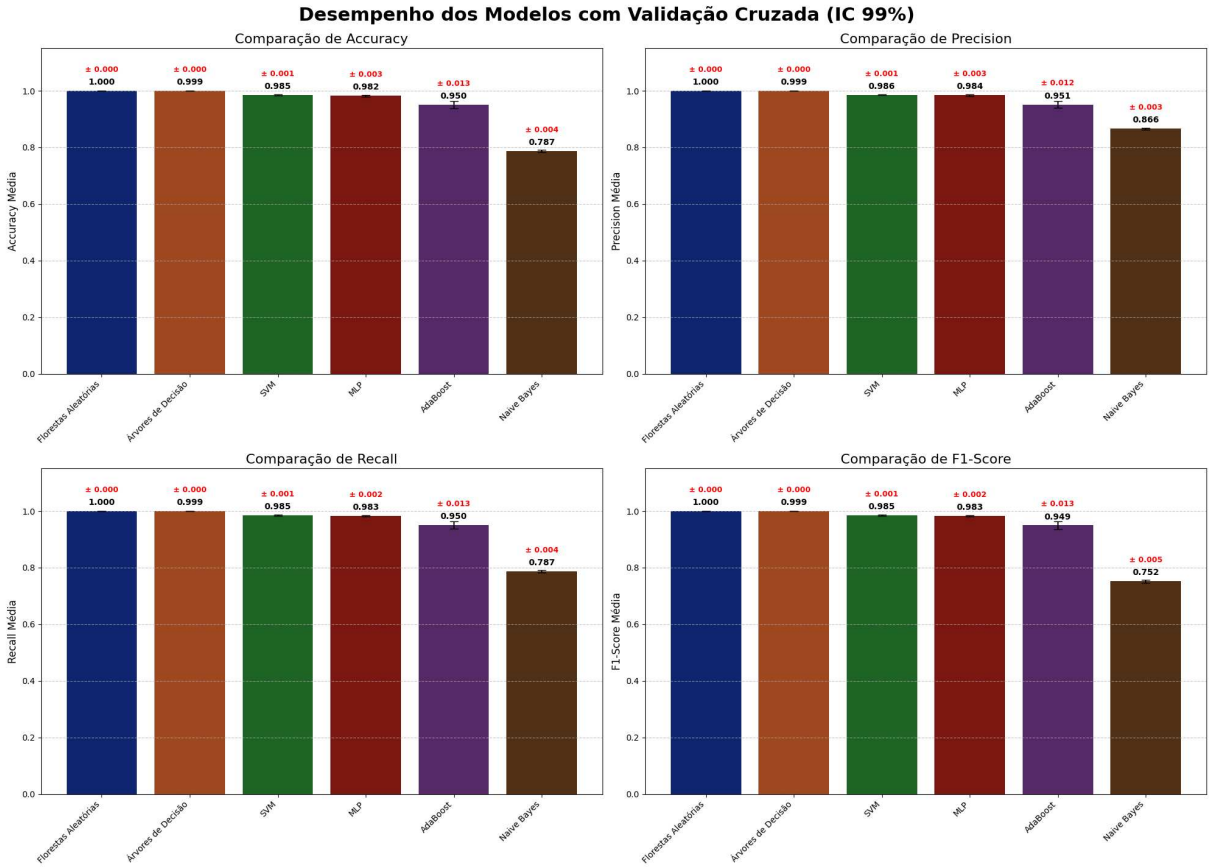

Figura 19 – Métricas para avaliação e intervalo de confiança dos modelos aplicados a fusão dos *datasets* CIC-2017, CIC-2018 e CIC-2023.

Fonte: Próprio autor.

Analisando a Figura 19, verifica-se que a maioria dos algoritmos obteve um excelente desempenho nos testes, com um nível de confiança de 99% da repetibilidade dos resultados,

garantindo que em quase 100% das vezes os valores médios obtidos serão semelhantes aos apresentados no gráfico. A Figura 20 mostra as matrizes de confusão referentes ao experimento final, nela, observa-se que, assim como indicado no gráfico anterior, quase todos os algoritmos tiveram um desempenho excepcional, tendo resultados abaixo dos outros apenas nos algoritmos Naive Bayes e AdaBoost. Contudo, o restante dos algoritmos, como SVM, Árvores de Decisão e Florestas Aleatórias, tiveram resultados praticamente impecáveis, demonstrando uma excelente generalização dos dados. Observa-se, ainda, que os quantitativos de falsos positivos e falsos negativos, em relação à amostra de testes, demonstram alta eficácia na detecção. Esse desempenho é essencial em cenários reais, pois evita classificações incorretas e, consequentemente, prejuízos de qualquer natureza, contribuindo diretamente para a confiabilidade do sistema.

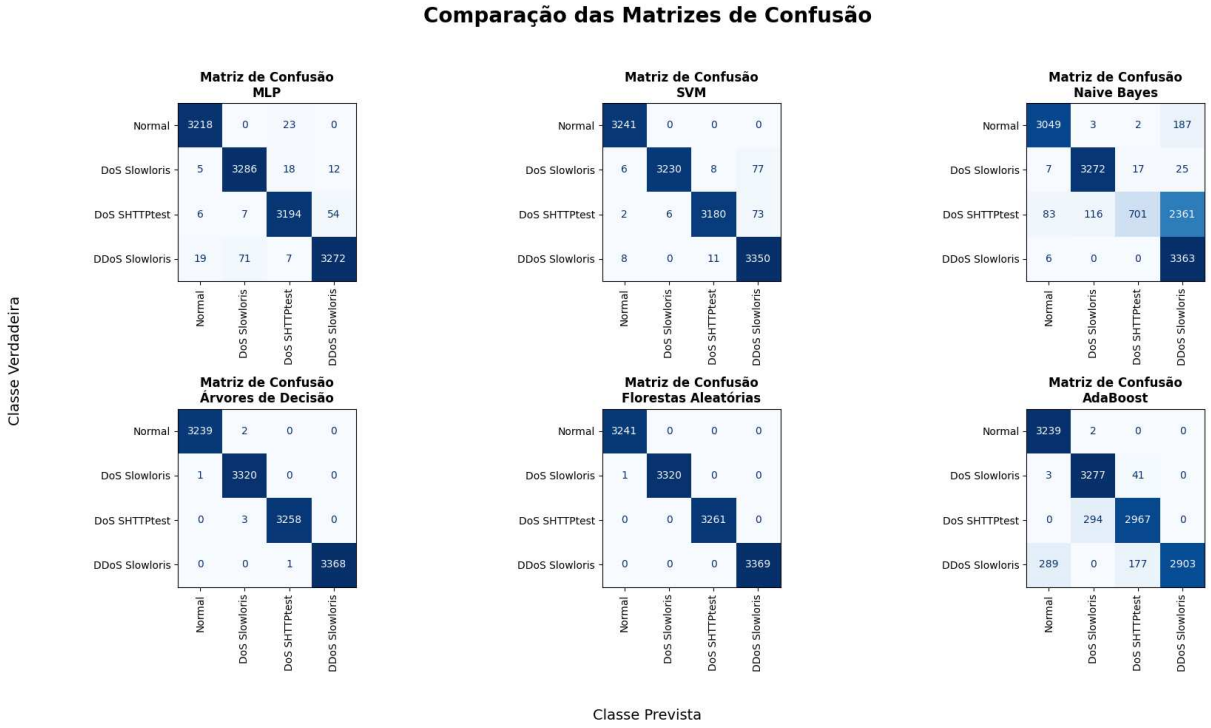

Figura 20 – Matriz de confusão dos modelos aplicados à fusão dos *datasets* CIC-2017, CIC-2018 e CIC-2023.

Fonte: Próprio autor.

A seguir, a Figura 21 mostra a curva ROC e valida os modelos implementados, sugerindo que temos uma configuração e modelos de ML robustos para implementações no mundo real, evidenciando que o classificador possui excelente capacidade de discriminação em diferentes limiares de decisão. Observe que o gráfico apresenta a curva ROC para todos os algoritmos implementados, sendo, mais uma vez, o Naive Bayes o que obteve o menor desempenho.

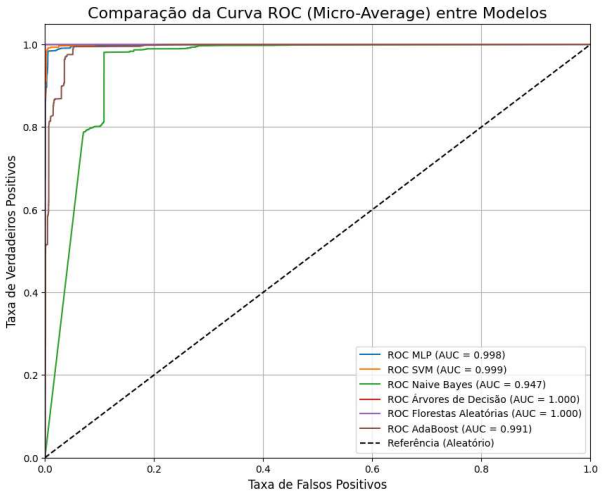

Figura 21 – Curva ROC dos modelos aplicados a fusão dos *datasets* CIC-2017, CIC-2018 e CIC-2023.

Fonte: Próprio autor.

6.1.6 Limitações do Trabalho

Apesar dos resultados promissores, o estudo se limitou aos ataques *Slowloris* e *SlowHTTPTest* presentes nos *datasets*. A eficácia contra uma gama ainda maior de ataques de camada de aplicação não foi testada. A extração de *features* com o *CICFlowMeter*, embora poderosa, foi um passo realizado *offline*. Uma implementação totalmente em tempo real exigiria a integração dessa etapa no próprio dispositivo de borda, o que impõe desafios computacionais adicionais. Esses desafios são plenamente viáveis de serem superados, tanto pelo fato de estarmos utilizando um dispositivo simples, como o Raspberry Pi 3b+, quanto pela existência de alternativas com capacidades computacionais significativamente superiores, como o Raspberry Pi 5 e outros dispositivos mais robustos e adequados ao uso de IA, como o ATLAS da Huawei, em suas diversas versões. Por fim, destaca-se que os testes foram realizados em ambientes controlados, e a validação em redes de produção com tráfego real ainda precisa ser conduzida.

7 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho teve como objetivo principal o desenvolvimento de uma plataforma de ML para a detecção e mitigação de ataques DoS e DDoS de camada de aplicação, visando ao aumento de confiabilidade de sistemas IoT em nós de borda.

Foi elaborada uma metodologia diferenciada para a aplicação de algumas técnicas de análise de séries temporais, aplicadas e avaliadas. Concluímos que, para todas as técnicas aplicadas, foram obtidos excelentes resultados com a maioria dos algoritmos aplicados no mesmo cenário. Ou seja, testando com os modelos treinados nos 80% dos dados e testados com os 20% restantes do mesmo *dataset*. Entretanto, ao inferir uma validação temporal dos modelos, identificamos que as técnicas não responderam tão bem ao aplicar dados de cenários diferentes (outro *dataset*) aos modelos treinados para a maioria dos casos.

Contudo, a Configuração 5, que avalia a aplicação de Médias Móveis para suavização dos dados e Entropia de Shannon, obteve resultados excelentes na avaliação temporal, chegando a 99% de acurácia, sensibilidade, precisão e F1-Score nos melhores modelos, demonstrando melhor generalização dos modelos para análise de tráfego de dados de redes para classificação de ataques DoS e DDoS de camada de aplicação.

Adicionalmente, foi observado que os modelos, além de apresentarem resultados de classificação e detecção excelentes, destacaram-se também pelo desempenho computacional, sendo testados dentro de um dispositivo de borda simples e rodando sem esforço computacional máximo do dispositivo.

Dessa forma, conclui-se que este trabalho, de forma geral, obteve resultados significativos e contribuições científicas relevantes para o avanço de estudos na área, além de possibilitar a aplicação dos modelos desenvolvidos na plataforma proposta de forma prática.

Como trabalhos futuros, pretende-se expandir a plataforma para detectar outros tipos de ataques de camada de aplicação. Além disso, planeja-se investir na otimização do *pipeline* de extração de *features* para execução direta em *hardware* de borda, buscando desempenho máximo a partir de configurações e técnicas de *software*. Pretende-se ainda explorar a aplicação de aprendizado federado, permitindo que múltiplos dispositivos de borda colaborem no treinamento de um modelo global sem compartilhar os dados de tráfego, aumentando a privacidade e diversidade dos dados de treinamento. Por fim, propõe-se a realização de estudos piloto em redes de produção, com o intuito de validar a eficácia da plataforma em cenários reais, considerando fatores como variabilidade do tráfego, interferências de rede e limitações de recursos.

REFERÊNCIAS

AI, X.; CHEN, H.; LIN, K.; WANG, Z.; YU, J. Nowhere to hide: Efficiently identifying probabilistic cloning attacks in large-scale rfid systems. **IEEE Transactions on Information Forensics and Security**, IEEE, v. 16, p. 714–727, 2020.

ALAHMADI, A. A.; ALJABRI, M.; ALHAIDARI, F.; ALHARTHI, D. J.; RAYANI, G. E.; MARGHALANI, L. A.; ALOTAIBI, O. B.; BAJANDOUH, S. A. Ddos attack detection in iot-based networks using machine learning models: a survey and research directions. **Electronics**, MDPI, v. 12, n. 14, p. 3103, 2023.

ALMEGHLEF, S. M.; AL-GHAMDI, A. A.-M.; RAMZAN, M. S.; RAGAB, M. Application layer-based denial-of-service attacks detection against iot-coap. **Electronics**, MDPI, v. 12, n. 12, p. 2563, 2023.

ASHTON, K. *et al.* That 'internet of things' thing. **RFID journal**, Hauppauge, New York, v. 22, n. 7, p. 97–114, 2009.

ASLAN, ; AKTUğ, S. S.; OZKAN-OKAY, M.; YILMAZ, A. A.; AKIN, E. A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions. **Electronics**, v. 12, n. 6, 2023. ISSN 2079-9292. Disponível em: <https://www.mdpi.com/2079-9292/12/6/1333>.

ASLAN, ; YILMAZ, A. A. A new malware classification framework based on deep learning algorithms. **IEEE Access**, v. 9, p. 87936–87951, 2021.

BALA, B.; BEHAL, S. Ai techniques for iot-based ddos attack detection: Taxonomies, comprehensive review and research challenges. **Computer science review**, Elsevier, v. 52, p. 100631, 2024.

BELACHEW, H. M.; BEYENE, M. Y.; DESTA, A. B.; ALEMU, B. T.; MUSA, S. S.; MUHAMMED, A. J. Design a robust ddos attack detection and mitigation scheme in sdn-edge-iot by leveraging machine learning. **IEEE Access**, v. 13, p. 10194–10214, 2025.

BISHOP, C. M.; NASRABADI, N. M. **Pattern recognition and machine learning**. [S.l.]: Springer, 2006. v. 4.

Canadian Institute for Cybersecurity (CIC). **CIC-IDS2018: Intrusion Detection Evaluation Dataset**. 2018. <https://www.unb.ca/cic/datasets/ids-2018.html>. Accessed: July 2025.

CHATFIELD, C. **The Analysis of Time Series: An Introduction**. 6. ed. Chapman and Hall/CRC, 2003. Disponível em: <https://www.taylorfrancis.com/books/9780429190717>.

CHEN, K.; ZHANG, S.; LI, Z.; ZHANG, Y.; DENG, Q.; RAY, S.; JIN, Y. Internet-of-things security and vulnerabilities: Taxonomy, challenges, and practice. **Journal of Hardware and Systems Security**, Springer, v. 2, p. 97–110, 2018.

(CIC), C. I. for C. **CICIDS 2017 Dataset**. 2017. Accessed: January 25, 2025. Disponível em: <https://www.unb.ca/cic/datasets/ids-2017.html>.

DAMGHANI, H.; DAMGHANI, L.; HOSSEINIAN, H.; SHARIFI, R. Classification of attacks on iot. In: **4th international conference on combinatorics, cryptography, computer science and computation**. [S.l.: s.n.], 2019. p. 245–255.

DATAPORT, I. **Bot-IoT Dataset**. 2018. Accessed: January 25, 2025. Disponível em: <https://ieee-dataport.org/documents/bot-iot-dataset>.

FERRAG MOHAMED AMINE E FRIHA, O. e. H. D. e. M. L. e. J. H. **Edge-IIoTset: Um novo conjunto de dados de segurança cibernética realista e abrangente de aplicativos de IoT e IIoT: aprendizagem centralizada e federada**. IEEE Dataport, 2022. Disponível em: <https://dx.doi.org/10.21227/mbc1-1h68>.

FILARETOV, G. F.; CHERVOVA, A. A. Fractal characteristics of the network traffic. In: IEEE. **2022 VI International Conference on Information Technologies in Engineering Education (Inforino)**. [S.l.], 2022. p. 1–4.

FILHO, E. de S. O. **Detecção de ataques Slow Loris e Slow HTTP Post utilizando Redes Neurais Artificiais**. Dissertação (Mestrado) — Instituto Militar de Engenharia (IME), Rio de Janeiro, RJ, Dezembro 2018. Disponível em: <http://www.defesacibernetica.ime.eb.br/pub/repositorio/2018_Edson_Souza.pdf>.

Fortune Business Insights. **Internet of Things (IoT) Market**. 2023. <https://www.fortunebusinessinsights.com/industry-reports/internet-of-things-iot-market-100307>. [Online; accessed 25-Nov-2023].

GOEL, M.; SINGH, S.; GARG, A.; ROY, N. R. Comparative study of ddos attacks & tools and their analysis. In: IEEE. **2023 International Conference on IoT, Communication and Automation Technology (ICICAT)**. [S.l.], 2023. p. 1–8.

GUERRA-MANZANARES, A.; MEDINA-GALINDO, J.; BAHSI, H.; NÕMM, S. Medbiot: Generation of an iot botnet dataset in a medium-sized iot network. In: **ICISSP**. [S.l.: s.n.], 2020. p. 207–218.

HASSIJA, V.; CHAMOLA, V.; SAXENA, V.; JAIN, D.; GOYAL, P.; SIKDAR, B. A survey on iot security: application areas, security threats, and solution architectures. **IEEe Access**, IEEE, v. 7, p. 82721–82743, 2019.

IBM. **Decision Trees: How They Work and How They're Used**. 2024. Accessed: 2025-02-17. Disponível em: <https://www.ibm.com/topics/decision-trees>.

IBM. **Naive Bayes: Understanding and Applications**. 2024. Accessed: 2025-02-17. Disponível em: <https://www.ibm.com/topics/naive-bayes>.

IoT Analytics. **The Number of Connected IoT Devices Continues to Grow**. 2024. Accessed: 2024-01-15. Disponível em: <https://iot-analytics.com/number-connected-iot-devices/>.

JIA, Y.; ZHONG, F.; ALRAWAIS, A.; GONG, B.; CHENG, X. Flowguard: An intelligent edge defense mechanism against iot ddos attacks. **IEEE Internet of Things Journal**, v. 7, n. 10, p. 9552–9562, 2020.

KANG, H.; AHN, D. H.; LEE, G. M.; YOO, J. D.; PARK, K. H.; KIM, H. K. **IoT network intrusion dataset**. IEEE Dataport, 2019. Disponível em: <https://dx.doi.org/10.21227/q70p-q449>.

KHANAM, S.; AHMEDY, I. B.; IDRIS, M. Y. I.; JAWARD, M. H.; SABRI, A. Q. B. M. A survey of security challenges, attacks taxonomy and advanced countermeasures in the internet of things. **IEEE access**, IEEE, v. 8, p. 219709–219743, 2020.

KIM, Y.-J.; JEONG, M.-A. Efficient anomaly detection through confidence interval estimation based on time series analysis. **The International Journal of Advanced Smart Convergence**, v. 4, n. 2, p. 46–53, 2015.

KRISHNA, R. R.; PRIYADARSHINI, A.; JHA, A. V.; APPASANI, B.; SRINIVASULU, A.; BIZON, N. State-of-the-art review on iot threats and attacks: Taxonomy, challenges and solutions. **Sustainability**, v. 13, n. 16, 2021. ISSN 2071-1050. Disponível em: <https://www.mdpi.com/2071-1050/13/16/9463>.

KSHIRSAGAR, D.; KUMAR, S. A feature reduction based reflected and exploited ddos attacks detection system. **Journal of Ambient Intelligence and Humanized Computing**, Springer, v. 13, n. 1, p. 393–405, 2022.

KUMAR, A.; SINGH, D. Detection and prevention of ddos attacks on edge computing of iot devices through reinforcement learning. **International Journal of Information Technology**, Springer, v. 16, n. 3, p. 1365–1376, 2024.

KUMARI, P.; JAIN, A. K. Timely detection of ddos attacks in iot with dimensionality reduction. **Cluster Computing**, Springer, p. 1–19, 2024.

LIU, Z.; HU, C.; SHAN, C. Riemannian manifold on stream data: Fourier transform and entropy-based ddos attacks detection method. **Computers & Security**, Elsevier, v. 109, p. 102392, 2021.

MAHJABIN, T.; XIAO, Y.; LI, T.; CHEN, C. P. Load distributed and benign-bot mitigation methods for iot dns flood attacks. **IEEE Internet of Things Journal**, IEEE, v. 7, n. 2, p. 986–1000, 2019.

MISHRA, N.; PANDYA, S. Internet of things applications, security challenges, attacks, intrusion detection, and future visions: A systematic review. **IEEE Access**, IEEE, v. 9, p. 59353–59377, 2021.

MORAES, J. M. L. **DDSHP: Um Sistema para a Detecção de DDoS em IoT Baseado no Parâmetro de Hurst e SDN**. Dissertação (Dissertação de Mestrado) — Universidade Federal do Ceará (UFC), Fortaleza, 2023. Orientador: Prof. Dr. Arthur de Castro Callado.

MORETTIN, P. A.; TOLOI, C. M. **Análise de séries temporais: modelos lineares univariados**. [S.l.]: Editora Blucher, 2018.

MOUSTAFA, Z.; TURNBULL, B.; CHOO, K.-K. R. **A Bot-IoT Dataset for Benchmarking Machine Learning Models**. Zenodo, 2020. Disponível em: <https://zenodo.org/record/4743746>.

NETO, E. C. P.; DADKHAH, S.; FERREIRA, R.; ZOHOURIAN, A.; LU, R.; GHORBANI, A. A. Ciciot2023: A real-time dataset and benchmark for large-scale attacks in iot environment. **Sensors**, v. 23, n. 13, 2023. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/23/13/5941>.

PRASAD, A.; CHANDRA, S. Vmfcvd: an optimized framework to combat volumetric ddos attacks using machine learning. **Arabian Journal for Science and Engineering**, Springer, v. 47, n. 8, p. 9965–9983, 2022.

RAUBITZEK, S.; NEUBAUER, T. Combining measures of signal complexity and machine learning for time series analyis: A review. **Entropy**, v. 23, n. 12, 2021. ISSN 1099-4300. Disponível em: <https://www.mdpi.com/1099-4300/23/12/1672>.

SASI, T.; LASHKARI, A. H.; LU, R.; XIONG, P.; IQBAL, S. A comprehensive survey on iot attacks: Taxonomy, detection mechanisms and challenges. **Journal of Information and Intelligence**, v. 2, n. 6, p. 455–513, 2024. ISSN 2949-7159. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2949715923000793>.

SATHYANARAYANAN, S.; TANTRI, B. R. Confusion matrix-based performance evaluation metrics. **African Journal of Biomedical Research**, v. 27, n. 4S, p. 4023–4031, 2024.

SENGUPTA, J.; RUJ, S.; Das Bit, S. A comprehensive survey on attacks, security issues and blockchain solutions for iot and iiot. **Journal of Network and Computer Applications**, v. 149, p. 102481, 2020. ISSN 1084-8045. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1084804519303418>.

SHARAFALDIN, I.; LASHKARI, A. H.; GHORBANI, A. A. *et al.* Toward generating a new intrusion detection dataset and intrusion traffic characterization. **ICISSp**, v. 1, n. 2018, p. 108–116, 2018.

SILVA, R. da. Entropia e a teoria da informação. **Revista Brasileira de Ensino de Ciência e Tecnologia**, IFPR - Instituto Federal do Paraná, v. 11, n. 1, p. 45–60, 2018. Disponível em: <https://www.ifpr.edu.br/revistatecnologia/entropia-teoria-informacao.pdf>. Acesso em: jul. 2025.

SOLANKI, M.; CHAUDHARI, S. Ddos attack forensics pattern identification using entropy and hurst coefficient based fusion model. In: **2023 IEEE 8th International Conference for Convergence in Technology (I2CT)**. [S.l.: s.n.], 2023. p. 1–7.

SYSTEMS, I. N. **How to Analyze and Reduce the Risk of DDoS Attacks**. 2023. Accessed: 2023-10-01. Disponível em: <https://www.netscout.com>.

TSAY, R. S. **Analysis of Financial Time Series**. 2. ed. John Wiley & Sons, 2005. Disponível em: <https://www.wiley.com/en-us/Analysis+of+Financial+Time+Series%2C+2nd+Edition-p-9780471690740>.

TSIKNAS, K.; TAKETZIS, D.; DEMERTZIS, K.; SKIANIS, C. Cyber threats to industrial iot: A survey on attacks and countermeasures. **IoT**, v. 2, n. 1, p. 163–186, 2021. ISSN 2624-831X. Disponível em: <https://www.mdpi.com/2624-831X/2/1/9>.

UDDIN, R.; KUMAR, S. A.; CHAMOLA, V. Denial of service attacks in edge computing layers: Taxonomy, vulnerabilities, threats and solutions. **Ad Hoc Networks**, Elsevier, v. 152, p. 103322, 2024.

VACCARI, I.; CHIOLA, G.; AIELLO, M.; MONGELLI, M.; CAMBIASO, E. Mqttset, a new dataset for machine learning techniques on mqtt. **Sensors (Basel)**, v. 20, n. 22, p. 6578, Nov 2020.

VALDOVINOS, I. A.; PÉREZ-DÍAZ, J. A.; CHOO, K.-K. R.; BOTERO, J. F. Emerging ddos attack detection and mitigation strategies in software-defined networks: Taxonomy, challenges and future directions. **Journal of Network and Computer Applications**, Elsevier, v. 187, p. 103093, 2021.

WIENS, F.; ZITZMANN, A. Predation on a wild slow loris (nycticebus coucang) by a reticulated python (python reticulatus). **Folia Primatologica**, v. 70, n. 6, p. 362–364, 1999.

XIAO, Y.; JIA, Y.; LIU, C.; CHENG, X.; YU, J.; LV, W. Edge computing security: State of the art and challenges. **Proceedings of the IEEE**, IEEE, v. 107, n. 8, p. 1608–1631, 2019.

YU, L.; QI, D. Hölder exponent and multifractal spectrum analysis in the pathological changes recognition of medical ct image. In: **2011 Chinese Control and Decision Conference (CCDC)**. [S.l.: s.n.], 2011. p. 2040–2045.

YUNGAICELA-NAULA, N. M.; VARGAS-ROSALES, C.; PEREZ-DIAZ, J. A. Sdn-based architecture for transport and application layer ddos attack detection by using machine and deep learning. **IEEE Access**, IEEE, v. 9, p. 108495–108512, 2021.