UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Giovanna Oliveira Martins

Classificação Hierárquica Aplicada a Sistemas de Detecção de Intrusões em Redes IoT

Uberlândia, Brasil

2025

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Giovanna Oliveira Martins

Classificação Hierárquica Aplicada a Sistemas de Detecção de Intrusões em Redes IoT

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Rodrigo Sanches Miani

Universidade Federal de Uberlândia – UFU

Faculdade de Computação

Bacharelado em Ciência da Computação

Uberlândia, Brasil

2025

Giovanna Oliveira Martins

Classificação Hierárquica Aplicada a Sistemas de Detecção de Intrusões em Redes IoT

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Ciência da Computação.

Trabalho aprovado. Uberlândia, Brasil, 01 de novembro de 2016:

Prof. Dr. Rodrigo Sanches Miani
Orientador

Profa. Dra. Elaine Ribeiro de Faria Paiva

Prof. Dr. Daniel Duarte Abdala

Uberlândia, Brasil
2025

Dedico este trabalho aos meus pais, os dois maiores cientistas que já conheci, pelo amor incondicional e pelo apoio constante, que tornaram este sonho possível.

Agradecimentos

Agradeço, em primeiro lugar, às forças divinas, pela orientação e pela força concedida ao longo de toda a minha trajetória acadêmica.

Agradeço ao meu pai, Luiz Gustavo, pela motivação constante, pelo cuidado e pelo apoio incondicional, especialmente nos momentos mais difíceis. Obrigada por me guiar pela vida, seu amor e dedicação foram fundamentais para que eu chegasse até aqui.

À minha família, pelo carinho e pela confiança depositada em mim. Em especial, à minha avó, Maria de Lourdes, que gentilmente se dispôs a ser a primeira leitora deste trabalho.

Ao meu namorado, João Pedro, que esteve comigo desde o primeiro semestre, por acreditar em mim e me incentivar a persistir até o fim.

Aos colegas e amigos que estiveram ao meu lado durante esse percurso, tornando a caminhada mais divertida e significativa.

Agradeço ao professor orientador Rodrigo Miani por todo o apoio, dedicação e comprometimento, tornando este trabalho viável.

Aos professores Elaine Paiva e Daniel Abdala, pela disponibilidade e contribuição como membros da banca examinadora.

Ao corpo docente da Faculdade de Computação, pelo empenho e excelência no ensino, que despertaram em mim a paixão pela Ciência da Computação.

E, especialmente, agradeço à minha mãe, Gina Maira, por sempre acreditar no meu potencial e preencher minha vida com amor, sonhos e sabedoria, apoiando-me em cada momento. Obrigada por ser minha maior inspiração, não apenas como mãe, mas também como professora e cientista. Nada disso teria sido possível sem você.

—

"Quanto mais estudo, mais sinto que minha mente nisso é insaciável."

— Ada Lovelace

Resumo

A evolução das Tecnologias de Informação e Comunicação trouxe tanto benefícios quanto riscos, especialmente com o surgimento da Internet das Coisas (IoT), que aumentou as vulnerabilidades a ataques cibernéticos. Neste contexto, Sistemas de Detecção de Intrusão (IDS), que utilizam aprendizado de máquina para classificar tráfego malicioso, são essenciais para identificar ataques em redes IoT. Contudo, classificadores planos tradicionais, sejam binários ou multiclasse, apresentam limitações no *trade-off* entre alta sensibilidade na filtragem de intrusões e granularidade na identificação de tipos de ataque. Diante disso, este trabalho propôs e investigou classificadores hierárquicos para detecção de intrusões em redes IoT, visando combinar a alta sensibilidade da classificação binária com a capacidade de distinção das formas de ataque proporcionadas pela classificação multiclasse. Utilizando o conjunto de dados CICIoT2023, dois modelos hierárquicos foram construídos e avaliados: Benigno-DDoS-Multiataque (BDM) e Benigno-Recon-DDoS-Multiataque (BRDM). Os classificadores hierárquicos combinaram com sucesso as qualidades de ambos os classificadores planos, com o BRDM aprimorando a detecção ao adicionar um nível específico para ataques *Recon*, reduzindo erros de predição e superando o modelo multiclasse na detecção das classes como *Spoofing*, *Brute Force* e *Web-Based*. Conclui-se que a classificação hierárquica é uma abordagem promissora para IDS em redes IoT, superando as limitações dos classificadores planos e contribuindo para a interpretabilidade e robustez dos sistemas.

Palavras-chave: Classificação Hierárquica, Sistemas de Detecção de Intrusão (IDS), Internet da Coisas (IoT), Cibersegurança, Aprendizado de Máquina

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

Tecnologias de Informação e Comunicação (TICs) estão em um contínuo e acelerado processo de desenvolvimento. A Internet das Coisas (IoT) está diretamente inserida nesse cenário de avanço das TICs, conectando dispositivos físicos à Internet e criando um ambiente digital mais integrado. No entanto, essa conectividade também amplia as vulnerabilidades, facilitando a ação de cibercriminosos, que podem explorar falhas nesses dispositivos para realizar ataques em larga escala, comprometendo redes inteiras. Nesse cenário, sistemas de segurança, como os Sistemas de Detecção de Intrusão (IDS), têm mostrado ser essenciais (HAFEZIAN; NADERAN; JADERYAN, 2024). Um IDS tem o objetivo de detectar tentativas de ataque em um ambiente computacional a partir do monitoramento e análise de eventos que ocorrem nesse ambiente, frequentemente recorrendo a algoritmos de aprendizado de máquina para classificar o tráfego como normal ou malicioso.

A área de aprendizado de máquina visa capacitar sistemas computacionais a aprender e a realizar previsões com base em dados disponíveis. Isso é alcançado por meio de algoritmos que identificam padrões e características significativas no conjunto de dados, construindo um modelo de classificação. No contexto do aprendizado supervisionado, são construídos classificadores que aprendem a partir de dados rotulados, visando prever rótulos reais com precisão, mesmo em casos em que o conjunto de treinamento pode ser limitado.

No âmbito de sistemas de detecção de intrusão, o aspecto mais crítico consiste em evitar que tráfego malicioso seja classificado como benigno, uma vez que tal falha comprometeria diretamente a integridade e a segurança da rede (JOO; HONG; HAN, 2003). Ademais, a correta classificação do tráfego de rede desempenha um papel fundamental não apenas na detecção de atividades maliciosas, mas também na identificação dos tipos específicos de ataques em execução. Essa distinção é essencial, uma vez que possibilita o planejamento de estratégias de defesa mais direcionadas e específicas ao ataque. Para enfrentar esses desafios, diferentes paradigmas de classificação são empregados, como a classificação binária e a multiclasse. A primeira distingue entre duas categorias principais, como tráfego benigno e malicioso, sendo particularmente útil na filtragem de ataques no geral. Por sua vez, a classificação multiclasse amplia esse escopo ao discriminar diferentes categorias de ataques, permitindo compreender a natureza específica da ameaça, ainda que tal abordagem acarrete maior complexidade e risco de erros de detecção, conforme o número de classes aumenta.

Dessa forma, embora os métodos tradicionais de classificação binária e multiclasse,

também conhecidos como classificadores planos, tenham desempenhado um papel importante no avanço dos sistemas de detecção de intrusão, eles apresentam limitações quando aplicados a problemas complexos, como os ataques em redes IoT, conforme evidenciado pelas investigações discutidas na Seção 2.2.1. A classificação binária, apesar de eficiente para separar tráfego malicioso de benigno, não fornece informações detalhadas sobre o tipo específico de ataque. Já a classificação multiclasse, embora capaz de distinguir entre diferentes categorias de ataques, sofre com o aumento de complexidade à medida que o número de classes aumenta, o que pode diminuir a sensibilidade do modelo na filtragem de ataques.

Nesse contexto, a classificação hierárquica surge como uma alternativa promissora à classificação plana tradicional, pois organiza as classes em uma estrutura hierárquica, explorando explicitamente as relações estruturais entre as classes (UDDIN et al., 2025). Essa abordagem permite, por exemplo, que o sistema primeiramente diferencie o tráfego benigno do malicioso e, em seguida, refine progressivamente a detecção em níveis mais específicos, categorizando os tipos de ataques de acordo com suas características e similaridades. Assim, esse método combina os benefícios de ambos os modelos binário e multiclasse, decompondo o problema de classificação em subproblemas menores e mais manejáveis.

Nesse sentido, o propósito deste estudo é investigar abordagens de classificação hierárquica para a criação de modelos de detecção de intrusão em ambientes IoT. Considerando a complexidade e a diversidade dos ataques, busca-se avaliar se a utilização de estruturas hierárquicas pode oferecer ganhos em robustez e interpretabilidade em relação aos modelos tradicionais de classificação binária e multiclasse. Para tal, os modelos serão treinados e testados com o conjunto de dados "CICIoT2023", um extenso conjunto de dados de ataques em redes IoT desenvolvido pelo *Canadian Institute for Cybersecurity* (NETO et al., 2023).

Ao propor e analisar modelos hierárquicos aplicados ao conjunto de dados CICIoT2023, este trabalho pretende não apenas preencher lacunas existentes na literatura, mas também fornecer *insights* práticos para o desenvolvimento de sistemas de segurança mais eficazes e adaptados à realidade dinâmica das redes IoT.

1.1 Motivação

De acordo com o Panorama de Ameaças para a América Latina 2024, o Brasil ocupa a posição de segundo país mais atacado no mundo, registrando 1.379 ataques cibernéticos por minuto, o que reforça a importância do investimento em cibersegurança no país (CNN Brasil, 2024b). Além disso, o aumento de 8,86% no número de ataques entre o primeiro e o segundo semestre de 2023 evidencia a crescente sofisticação das ameaças

digitais e a vulnerabilidade das redes nacionais (CNN Brasil, 2024a). Para a sociedade, esses ataques comprometem a segurança de dados pessoais e institucionais, gerando prejuízos financeiros e riscos à privacidade. Esses números reforçam a importância do estudo e desenvolvimento de tecnologias capazes de mitigar essas ameaças, entre elas, algoritmos de aprendizado de máquina aplicados a IDS, que podem melhorar a identificação de ameaças em tempo real.

Paralelamente, a Internet das Coisas tem demonstrado um crescimento contínuo. O número de dispositivos conectados atingiu a marca de 16,6 bilhões no final de 2023, representando um aumento de 15% em relação ao ano anterior, com projeções que apontam para 40 bilhões até 2030 (IoT Analytics, 2025). Contudo, essa expansão vem acompanhada de riscos: em 2023, observou-se um aumento de 400% em ataques de *malware* direcionados a dispositivos IoT em comparação com 2022. Assim, a ampliação da superfície de ataque torna ainda mais crítica a necessidade de sistemas eficazes de detecção de intrusão especificamente voltados para redes IoT, capazes de identificar e mitigar vulnerabilidades, proteger a integridade dos dados e garantir a confiança necessária para a expansão segura desse ecossistema.

Nesse cenário, estudos recentes têm demonstrado avanços significativos por meio da aplicação de técnicas de aprendizado de máquina, empregando principalmente classificadores binários e multiclasse (KUMAR; RASTOGI; RANGA, 2024; HAFEZIAN; NADERAN; JADERYAN, 2024; ESTEVES, 2025). No entanto, observa-se que a maioria desses trabalhos concentra-se em abordagens planas que, embora úteis, apresentam suas próprias limitações. A classificação hierárquica, por sua vez, ainda é pouco explorada nesse domínio, o que evidencia uma lacuna de pesquisa a ser preenchida. Tal cenário reforça a relevância de investigar o potencial dessa abordagem para aprimorar a detecção de intrusões em ambientes IoT.

Dessa maneira, a motivação desta pesquisa é contribuir para o avanço das técnicas de identificação de atividades maliciosas em redes IoT por meio da exploração da classificação hierárquica, uma abordagem ainda pouco investigada nesse domínio.

1.2 Objetivos

O objetivo principal deste estudo é construir um classificador hierárquico para a detecção de intrusões em redes IoT, de modo a combinar a eficiência da filtragem de ataques obtida com classificadores binários e a capacidade de distinção entre múltiplos tipos de ataques, característica da classificação multiclasse. Ao longo desse processo, almeja-se atingir os seguintes objetivos específicos:

- Investigar diferentes formas de organização da hierarquia de classificadores, com o

intuito de reduzir a confusão entre classes e aprimorar a detecção de ataques;

- Validar o desempenho preditivo e computacional do modelo proposto para o conjunto de dados CICIoT2023;
- Comparar seus resultados em relação a abordagens planas binária e multiclasse;
- Analisar os erros de classificação em cada nível hierárquico, identificando classes mais problemáticas e tipos de ataques que permanecem desafiadores;
- Avaliar a contribuição da abordagem hierárquica para a interpretabilidade e robustez dos sistemas de detecção de intrusão em ambientes IoT.

1.3 Método

O processo metodológico adotado neste estudo pode ser estruturado em quatro etapas principais. Primeiramente, realiza-se a coleta e a análise do conjunto de dados CICIoT2023, com o objetivo de compreender suas características e potenciais desafios para a modelagem. Na sequência, aplica-se o pré-processamento e a transformação dos dados, incluindo procedimentos de limpeza, padronização e adequação dos atributos para garantir sua qualidade e compatibilidade com os algoritmos de aprendizado de máquina. Em um terceiro momento, são construídos e treinados tanto os classificadores planos (binário e multiclasse) quanto os classificadores hierárquicos propostos, assegurando condições experimentais equivalentes em termos de ambiente computacional e conjunto de treinamento. Por fim, os modelos são avaliados e comparados a partir de métricas padronizadas, de modo a verificar o alcance do objetivo principal e analisar as contribuições específicas da abordagem hierárquica frente às estratégias tradicionais. O desempenho preditivo dos algoritmos será avaliado por meio de métricas como acurácia, precisão, *recall*, *F1-score*, obtidas a partir da matriz de confusão, enquanto o desempenho computacional será analisado com base no tempo de treinamento, no tempo de teste e na memória RAM ocupada pelo modelo.

Para conduzir os testes deste estudo, os códigos foram desenvolvidos e executados no ambiente *Jupyter Notebook*, utilizando a linguagem *Python* (THOMAS et al., 2016). Para a implementação dos modelos, foram utilizadas as bibliotecas *Scikit-learn* (Scikit-learn Developers, 2025) e *XGBoost* (XGBoost Developers, 2022). Ademais, a manipulação e análise do conjunto de dados serão feitas por meio da biblioteca *Pandas* (MCKINNEY, 2010).

1.4 Organização da Monografia

A organização deste trabalho está estruturada da seguinte forma:

- Capítulo 2: apresenta a revisão bibliográfica, contemplando a fundamentação teórica e os trabalhos correlatos;
- Capítulo 3: descreve os passos seguidos para a execução do método proposto;
- Capítulo 4: expõe os resultados obtidos pelos experimentos;
- Capítulo 5 retoma os principais pontos abordados, destacando os *insights* alcançados, bem como as limitações do estudo e perspectivas para pesquisas futuras.

2 Revisão Bibliográfica

Este capítulo apresenta a revisão bibliográfica que fundamenta teoricamente o desenvolvimento deste trabalho. Serão abordados os principais conceitos e tecnologias relacionados ao tema, com o objetivo de oferecer uma base sólida para a compreensão do problema estudado. Além disso, serão analisados trabalhos correlatos que exploram abordagens semelhantes, permitindo identificar lacunas, inspirações e diferenciais que orientam a proposta desta pesquisa.

2.1 Fundamentação Teórica

Esta seção apresenta os principais conceitos e fundamentos necessários para o desenvolvimento deste trabalho. O objetivo é contextualizar os elementos centrais relacionados ao problema abordado e fornecer a base teórica que sustenta as metodologias empregadas.

Inicialmente, discute-se o conceito de Internet das Coisas (IoT), destacando sua crescente relevância e os desafios de segurança nesse ambiente distribuído. Em seguida, aborda-se a área de Cibersegurança, enfatizando as ameaças e vulnerabilidades que afetam redes IoT e que motivam a busca por soluções eficazes de monitoramento e defesa.

Na sequência, são apresentados os Sistemas de Detecção de Intrusão (IDS), descrevendo suas características, classificações e importância na identificação de atividades maliciosas. Posteriormente, explora-se o Aprendizado de Máquina, com foco em suas funcionalidades, metodologia e nos principais tipos de aprendizado.

Mais adiante, essa seção discute o conceito de Modelos de Classificação e suas possíveis abordagens, focando na distinção entre classificação plana e hierárquica. Em seguida, são apresentados os Algoritmos de Classificação utilizados neste trabalho, destacando seus princípios de funcionamento. Por fim, são detalhadas as Métricas de Avaliação adotadas nesta pesquisa, essenciais para medir o desempenho dos modelos e analisar sua eficácia frente ao problema de classificação.

2.1.1 Internet das Coisas

As Redes de Internet das Coisas (IoT) representam um ecossistema interconectado de dispositivos inteligentes, sensores, atuadores e plataformas computacionais que permitem a coleta, processamento e compartilhamento de dados e recursos em tempo real, agindo diante de situações e mudanças no ambiente. Segundo (BAHGA; MADISETTI, 2015), a IoT envolve tecnologias e arquiteturas que possibilitam a comunicação entre má-

quinas (M2M) e entre humanos e dispositivos, promovendo a automação e a eficiência em diversas aplicações.

O conceito da IoT surgiu em 1980, quando programadores da Carnegie Mellon University criaram um sistema que monitorava a disponibilidade de bebidas em uma máquina de refrigerantes, permitindo que pudessem verificar remotamente se valia a pena se deslocarem até a máquina. Embora esse conceito tenha surgido nessa época, o termo "Internet das Coisas"foi cunhado apenas em 1999 por Kevin Ashton, então diretor executivo do Auto-ID Labs no MIT (MADAKAM; RAMASWAMY; TRIPATHI, 2015). Desde então, diversos dispositivos físicos vêm sendo conectados à Internet, criando um ambiente digital mais integrado para melhorar a qualidade de vida e a automação em diversas áreas (NETO et al., 2023). No transporte, por exemplo, dispositivos IoT têm sido utilizados para coordenar o trânsito e prevenir acidentes (EURICO, 2023). Já na saúde, pacientes podem ser monitorados por uma rede de sensores interconectados para identificar eventuais situações de risco e agilizar a atuação médica (ALBANDES et al., 2019).

Nos últimos anos, a sociedade experimentou um aumento drástico nas conexões IoT, tendência que se manterá nos próximos anos (ALANI; DAMIANI, 2023). Contudo, esse ambiente digital cada vez mais integrado amplia a superfície de ataque disponível para agentes mal-intencionados, criando um cenário propício à exploração de vulnerabilidades. Em consequência, observa-se um crescimento significativo na ocorrência de ataques direcionados a dispositivos e redes IoT (NETO et al., 2023).

2.1.2 Cibersegurança

Cibersegurança é o conjunto de práticas, tecnologias e processos projetados para proteger redes, dispositivos e dados contra ameaças digitais. O principal objetivo da cibersegurança é garantir a proteção da informação através da aplicação da tríade Confidencialidade, Integridade e Disponibilidade (CID). A confidencialidade garante que apenas indivíduos autorizados tenham acesso às informações, a integridade assegura que os dados permaneçam inalterados e confiáveis e a disponibilidade garante que os sistemas e informações estejam acessíveis sempre que necessários (STEINBERG et al., 2023).

Ciberataques podem comprometer qualquer um desses princípios, causando desde o vazamento de informações sensíveis até a interrupção de serviços essenciais. Os principais métodos de ataques cibernéticos incluem *Denial of Service* (DoS e DDoS), *Abuse Tools*, *Logical Bombs*, *Sniffers*, *Trojan Horses*, *Virus*, *Worms*, *Spams* e *Botnets*. No ataque DoS, o invasor sobrecarrega um sistema com mensagens, impedindo o tráfego legítimo, e a ampliação desse ataque usando múltiplos dispositivos infectados é conhecida como ataques DDoS. *Abuse Tools* exploram vulnerabilidades de redes, enquanto *Logical Bombs* ativam códigos maliciosos diante de eventos específicos. *Sniffers* interceptam dados para capturar informações sensíveis e *Trojan Horses* se disfarçam de software útil para executar códigos

perigosos. Vírus infectam arquivos e precisam de interação humana para se espalhar, enquanto *Worms* se replicam automaticamente em redes. *Botnets* são redes de sistemas infectados usadas para espalhar malware, realizar ataques e roubar informações (LI; LIU, 2021).

Este trabalho foca em sete classes de ataques distintas: *DDoS*, *DoS*, *Recon*, *Web-Based*, *Brute Force*, *Spoofing* e *Mirai*. Como já explicado, *DDoS* e *DoS* são ataques que visam comprometer a disponibilidade de dispositivos IoT, exemplos incluem *SYN Flood*, *UDP Flood* e *HTTP Flood*. *Recon* refere-se à coleta de informações sobre um alvo, incluindo técnicas como *Port Scan* e *OS Scan*. Ataques *Web-Based* exploram vulnerabilidades em serviços *web*, como *SQL Injection* e *XSS*. *Brute Force* busca acesso a sistemas por tentativa repetitiva de credenciais, como no *Dictionary Brute Force*. *Spoofing* envolve falsificação de identidade para acessar sistemas ou roubar dados, incluindo *ARP* e *DNS spoofing*. *Mirai* é um *malware* que transforma dispositivos IoT em *botnets* para ataques DDoS massivos (NETO et al., 2023).

Dentre os ataques investigados, dois se destacam por terem características diferentes dos demais: *DDoS* e *Recon*. O ataque *DDoS* distingue-se por ser de natureza eminentemente volumétrica, caracterizando-se pela geração massiva de tráfego malicioso com o objetivo de sobrecarregar os recursos do alvo e comprometer sua disponibilidade (ZARGAR; JOSHI; TIPPER, 2013). Já o ataque *Recon* difere por se tratar de uma etapa preparatória no ciclo de um ataque, na qual o agente mal-intencionado realiza a coleta sistemática de informações sobre o alvo - como serviços em execução, portas abertas e sistemas operacionais utilizados - sem necessariamente executar ações destrutivas ou intrusivas de forma imediata (ALANI; DAMIANI, 2023).

2.1.3 Sistemas de Detecção de Intrusão

Os Sistemas de Detecção de Intrusão (IDS) são ferramentas de segurança projetadas para monitorar um ambiente computacional em busca de atividades suspeitas que possam violar as regras de segurança desse ambiente (BERGAMASCO et al., 2021). Seu objetivo principal é identificar acessos não autorizados, tentativas de exploração de vulnerabilidades e comportamentos anômalos, alertando administradores para mitigar possíveis ameaças. Portanto, ele funciona como uma segunda linha de defesa a invasões.

Existem dois tipos principais de IDS com base no posicionamento: *Host-based IDS* (HIDS) e *Network-based IDS* (NIDS). O sistema NIDS monitora o tráfego do segmento de rede no qual está inserido. A detecção é feita através da captura e análise dos cabeçalhos e conteúdos dos pacotes que passam pela rede. Já os sistemas HIDS monitoram apenas um único *host* na rede, eles são instalados diretamente em dispositivos individuais da rede, monitorando apenas o tráfego de entrada e saída desse equipamento (ZARPELÃO et al., 2017).

Outra forma de classificar Sistemas de Detecção de Intrusão é com base na metodologia de detecção, em que as duas formas principais são por anomalia e por assinatura. IDS Baseados em Assinatura comparam os dados de tráfego de rede com um banco de dados de assinaturas conhecidas de ameaças; se a atividade do sistema corresponder a uma assinatura na base de dados, o sistema emite um alerta indicando um possível ataque. IDS Baseados em Anomalia procuram por padrões que se desviam significativamente do que é considerado normal, através da comparação dos eventos com um modelo de comportamento esperado. Quando atividades saem desses limites predefinidos, o sistema os identifica como anomalias e emite alertas (PIETRO; MANCINI, 2008). Neste trabalho, investiga-se Sistemas de Detecção de Intrusão *Network-based* com detecção por anomalia.

Inicialmente, administradores de sistemas monitoravam manualmente atividades suspeitas em consoles, identificando possíveis intrusões de forma intuitiva, mas não escalável. Nos anos 1970 e 1980, a análise passou a ser feita por meio de *logs* de auditoria impressos, o que era demorado e servia mais para investigações pós-incidente do que para prevenção. Nos anos 1990, surgiram os primeiros sistemas de detecção de intrusão em tempo real, permitindo identificar e responder a ataques conforme ocorriam. Atualmente, os esforços se concentram em desenvolver sistemas mais eficazes para grandes redes, enfrentando desafios como novas ameaças e a constante evolução dos ambientes computacionais (MRCET, 2025). Nos últimos anos, o uso de Inteligência Artificial e Aprendizado de Máquina tem revolucionado os Sistemas de Detecção de Intrusão, tornando-os mais eficazes na identificação de ameaças desconhecidas e na resposta a ataques em tempo real (HORCHULHACK; SANTIN; VIEGAS, 2024).

2.1.4 Aprendizado de Máquina

Aprendizado de Máquina é uma subárea da Inteligência Artificial que se concentra no desenvolvimento de algoritmos que permitem aos computadores aprender padrões e tomar decisões através da experiência (MITCHELL, 1997). De acordo com Shalev-Shwartz e Ben-David (2014), o termo refere-se à detecção automatizada de padrões significativos em dados.

O Aprendizado de Máquina tem aplicações em diversas áreas, como, por exemplo, nas áreas de saúde e segurança. Ele desempenha um papel essencial no diagnóstico médico baseado em imagens, permitindo a identificação precoce de doenças como câncer por meio da análise automática de exames de imagem (GHASSEMI et al., 2020) (DESAI; SHAH, 2021) (ARDABILI et al., 2020). Já na segurança cibernética, técnicas de aprendizado de máquina são aplicadas na detecção de anomalias em redes e na identificação de possíveis ameaças, aumentando a capacidade de resposta contra ataques cibernéticos (MASEER et al., 2021) (AHMAD et al., 2022) (NETO et al., 2023).

O processo de desenvolvimento de um modelo de aprendizado de máquina segue

diversas etapas essenciais, as quais correspondem ao fluxo de Descoberta de Conhecimento em Bancos de Dados (KDD), um processo estruturado para a extração de conhecimento a partir de dados (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996). A primeira etapa é a seleção de dados, que envolve a escolha de um conjunto de dados relevante e representativo para o problema a ser resolvido. Em seguida, realiza-se o tratamento desses dados, incluindo limpeza, tratamento de valores ausentes e correção de inconsistências. A fase de transformação dos dados envolve normalização, codificação de variáveis categóricas e extração de características para melhorar o desempenho do modelo. A etapa seguinte consiste na mineração de dados, na qual se aplicam métodos e algoritmos para identificar padrões úteis dentro do conjunto de dados. Por fim, é feita a interpretação e a avaliação do desempenho do modelo por meio de métricas como acurácia, precisão, *recall* e *F1-score* (MOSAFI, 2022).

Existem dois tipos principais de aprendizado: supervisionado e não supervisionado. No aprendizado supervisionado, o modelo é treinado a partir de um conjunto de dados rotulado, ou seja, composto por exemplos cujas saídas foram previamente determinadas por especialistas (BISHOP, 2006). Esse processo permite que o modelo aprenda as relações entre as variáveis de entrada e suas respectivas saídas, de forma a generalizar esse conhecimento e realizar previsões sobre novos dados não rotulados provenientes de cenários reais. Durante o treinamento, o algoritmo ajusta iterativamente seus parâmetros internos para minimizar o erro entre as saídas previstas e as saídas corretas fornecidas nos dados de treino. Por sua vez, no aprendizado não supervisionado, utilizam-se dados não rotulados, empregando-se algoritmos capazes de identificar padrões e similaridades sem intervenção humana direta e treinamento prévio. Nessa abordagem, uma das principais tarefas é o agrupamento (*clustering*) de um conjunto de dados em subconjuntos de instâncias semelhantes, de modo que a homogeneidade entre os elementos de um mesmo grupo seja maximizada, enquanto a similaridade entre elementos de grupos distintos seja minimizada (FACELI et al., 2011).

O aprendizado supervisionado pode ser categorizado, de forma geral, em duas principais tarefas: classificação e regressão (FACELI et al., 2011). A classificação consiste em atribuir instâncias a categorias previamente definidas, por meio da identificação de padrões ou características específicas presentes nos dados, determinando, assim, a classe à qual cada instância pertence. Por sua vez, a regressão tem como objetivo modelar a relação entre variáveis de modo a estimar um resultado numérico. Assim, enquanto a classificação é utilizada para prever classes discretas, a regressão é aplicada quando a variável de saída assume valores contínuos (BELCIC; STRYKER, 2025). Neste trabalho, serão estudados exclusivamente modelos de classificação supervisionados.

2.1.5 Modelos de Classificação

Como apresentado anteriormente, modelos de classificação são técnicas de aprendizado de máquina que categorizam dados em classes predefinidas. Esses modelos de classificação podem ser divididos em classificação binária e classificação multiclasse (SHALEV-SHWARTZ; BEN-DAVID, 2014). A classificação binária refere-se a problemas em que existem apenas duas classes. Nesse caso, o objetivo do modelo é separar as instâncias em uma dessas duas categorias, como, por exemplo, em sistemas de detecção de fraude ou diagnósticos médicos que retornam a presença ou ausência de uma condição. Por outro lado, a classificação multiclasse envolve problemas em que há três ou mais classes possíveis. Nesse contexto, o modelo deve atribuir cada instância exatamente a uma dessas categorias, como, por exemplo, em sistemas de reconhecimento de dígitos manuscritos (classes de 0 a 9) ou na categorização de tipos de ataques em sistemas de detecção de intrusão (PH.D.; KAVLAKOGLU, 2025).

Uma outra forma de categorizar os modelos seria em classificação hierárquica ou plana (SILLA JR; FREITAS, 2011). A Figura 1 ilustra essas duas abordagens.

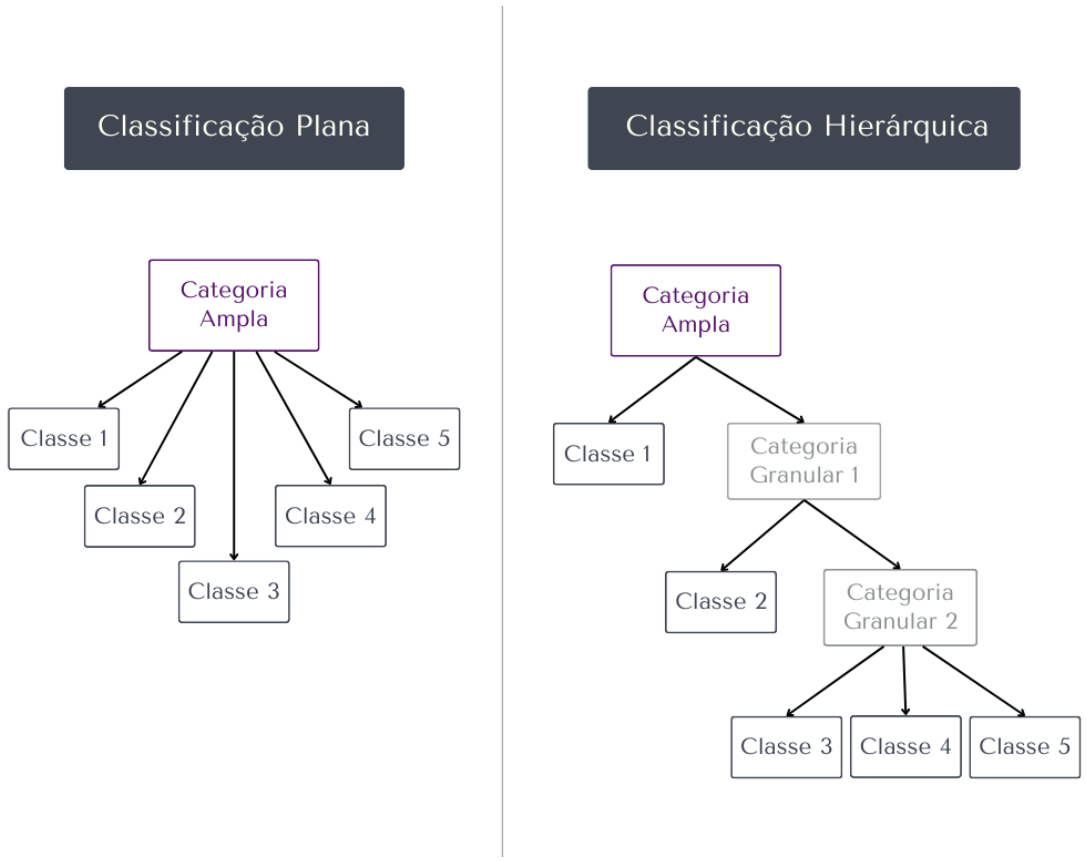

Figura 1 – Diferença entre classificação plana e hierárquica.

Na classificação plana (*flat classification*), as classes são tratadas como independentes e desprovidas de qualquer relação estrutural explícita. Esse paradigma inclui tanto

a classificação binária quanto a multiclasse tradicional, em que o modelo é treinado e avaliado sem considerar qualquer hierarquia pré-existente entre as classes. Quando aplicada a problemas intrinsecamente hierárquicos, essa abordagem prevê apenas as classes localizadas nos nós folha (classes alvo), assumindo implicitamente a atribuição de suas classes ancestrais. Ignorar essa estrutura hierárquica implica em perda de informação sobre as relações entre classes pai e filho, além de aumentar a complexidade do problema ao exigir classificadores capazes de distinguir um grande número de classes folha de forma isolada.

Em contrapartida, a classificação hierárquica é uma abordagem na qual as classes a serem previstas não são independentes, mas sim organizadas em uma estrutura hierárquica, geralmente representada por uma árvore ou um Grafo Acíclico Dirigido (DAG). Diferentemente dos problemas de classificação plana, a classificação hierárquica explora explicitamente as relações estruturais entre as classes. Ao incorporar essas relações, a classificação hierárquica permite reduzir a complexidade do problema, seja por meio de classificadores globais (que constroem um único modelo para toda a hierarquia) ou classificadores locais (que treinam modelos específicos para cada nó ou nível). Além disso, esse método possibilita um controle mais eficiente da propagação de erros, tornando-se uma estratégia fundamental para problemas em que a estrutura hierárquica das classes desempenha papel central.

2.1.6 Algoritmos de Classificação

Os algoritmos de classificação podem ser organizados segundo diferentes critérios, e uma das taxonomias mais influentes é a apresentada por Hastie, Tibshirani e Friedman (2009), que agrupa os métodos supervisionados de acordo com seus fundamentos estatísticos. Essa taxonomia destaca categorias como modelos lineares, métodos baseados em vizinhança, métodos baseados em árvores de decisão, máquinas de vetores de suporte e comitês de classificadores (*ensemble*).

Os modelos lineares utilizam combinações lineares dos atributos para realizar a separação entre classes, sendo exemplos comuns a Regressão Logística e a Análise Discriminante Linear. Os métodos baseados em vizinhança, como o k-Vizinhos Mais Próximos (KNN), classificam novas instâncias a partir da similaridade em relação aos exemplos de treinamento, considerando a proximidade no espaço de atributos. Os métodos baseados em árvores, como a Árvore de Decisão, constroem modelos de decisão a partir de divisões sucessivas dos dados, gerando regras interpretáveis de classificação. As Máquinas de Vetores de Suporte (SVM) buscam maximizar a margem de separação entre classes (HASTIE; TIBSHIRANI; FRIEDMAN, 2009). Por fim, os comitês de classificadores combinam múltiplos modelos de classificação com o objetivo de melhorar a acurácia e a robustez das previsões. Exemplos amplamente utilizados incluem *Random Forest*, *LightGBM* e *XGBoost* (NTAMWIZA; BWIRE, 2025).

Além dessas categorias, a literatura contemporânea também reconhece as redes neurais artificiais como uma família de modelos distinta (AGGARWAL, 2023a). As redes neurais são modelos inspirados no funcionamento do cérebro humano, compostos por camadas de neurônios artificiais interconectadose que processam e transmitem informações, permitindo que a rede neural aprenda padrões e relações complexas nos dados. Dentre suas arquiteturas, destacam-se as Redes Neurais Perceptron Multicamadas (*Multi-Layer Perceptron* - MLP), as Redes Neurais Convolucionais (*Convolutional Neural Networks* - CNN) e as Redes Neurais Recorrentes (*Recurrent Neural Networks* - RNN) (AGGARWAL, 2023b).

Os métodos baseados em árvores de decisão ocupam um papel central neste trabalho. A Árvore de Decisão é um modelo que divide o espaço dos dados em regiões distintas. Cada nó da árvore representa uma decisão baseada em um atributo do dado, e as folhas representam as classes finais (QUINLAN, 1996). Evoluções desses métodos deram origem aos comitês baseados em árvores, cujo objetivo é combinar múltiplos modelos considerados fracos a fim de produzir classificadores mais robustos e precisos. Nesse contexto, este trabalho concentra-se especificamente nos comitês de classificadores baseados em árvores de decisão, com ênfase nos algoritmos *Random Forest* e *XGBoost* (NTAMWIZA; BWIRE, 2025).

O *Random Forest*, por sua vez, constrói diversas árvores de decisão independentes a partir de subconjuntos aleatórios de dados e atributos. Nesse modelo, a predição final é obtida por meio de votação majoritária, na qual todas as predições individuais são contabilizadas e a classe mais frequente é selecionada como resultado final (BIAU; SCORNET, 2016). Em contraste, o *Extreme Gradient Boosting* (*XGBoost*) consiste em um algoritmo de *boosting* baseado em árvores que constrói o modelo de forma sequencial. A cada iteração, uma nova árvore é treinada com o objetivo de corrigir os erros cometidos pelas árvores anteriores, utilizando o gradiente da função de perda como guia para o ajuste dos parâmetros. Além disso, esse algoritmo incorpora otimizações relevantes, tais como mecanismos de regularização para evitar o sobreajuste (*overfitting*), paralelização do processo de treinamento e tratamento eficiente de valores ausentes, resultando em um modelo de elevada performance e escalabilidade (CHEN et al., 2015).

2.1.7 Métricas de Avaliação

Após a construção de um modelo de classificação, é essencial avaliar seu desempenho e compreender em quais situações ele apresenta maiores dificuldades de predição. As métricas de avaliação têm como objetivo mensurar a qualidade de um modelo de aprendizado de máquina, permitindo compreender sua capacidade de generalização e sua eficácia em relação à tarefa proposta. Em problemas de classificação, como a detecção de intrusões em redes IoT, tais métricas são fundamentais para identificar não apenas o desempenho

global do modelo, mas também sua habilidade em distinguir corretamente entre diferentes classes, especialmente em cenários de desbalanceamento de dados.

A avaliação de modelos de classificação baseia-se na comparação entre as previsões do modelo e os rótulos reais das amostras. A partir dessa comparação, quatro categorias principais podem ser definidas: Verdadeiro Positivo (VP), Verdadeiro Negativo (VN), Falso Positivo (FP) e Falso Negativo (FN). Considere um problema de classificação binária com duas classes: positiva e negativa. O VP representa o número de instâncias cuja classe real é positiva e que foram corretamente classificadas como positivas pelo modelo. Em outras palavras, são casos em que o modelo acertou ao prever o evento de interesse. O VN refere-se ao número de instâncias cuja classe real é negativa e que foram corretamente classificadas como negativas pelo modelo. Ou seja, são casos em que o modelo corretamente identificou a ausência do evento de interesse. O FP ocorre quando o modelo prevê a classe positiva, mas a classe real é negativa. Esse tipo de erro representa uma previsão incorreta da ocorrência do evento quando este não ocorreu. O FN ocorre quando o modelo prevê a classe negativa, mas a classe real é positiva. Esse erro indica que o modelo deixou de identificar um evento positivo que de fato ocorreu (MARIANO, 2021).

Essas quatro categorias formam a matriz de confusão, uma representação tabular que permite avaliar o desempenho de um classificador ao comparar as classes previstas pelo modelo com as classes reais observadas nos dados. Ela apresenta os acertos e erros em termos de contagens absolutas de ocorrências em cada categoria. A Tabela 1 apresenta a estrutura da matriz para problemas binários.

Tabela 1 – Matriz de confusão para problemas binários.

		Classe Predita	
		Positivo	**Negativo**
Classe Real	**Positivo**	Verdadeiro Positivo (VP)	Falso Negativo (FN)
	Negativo	Falso Positivo (FP)	Verdadeiro Negativo (VN)

Essa matriz permite uma análise detalhada dos erros cometidos pelo modelo, possibilitando compreender em quais situações ele tende a falhar. Em sistemas de detecção de intrusão, a matriz de confusão é especialmente importante para identificar falsos negativos (ataques não detectados ou erroneamente classificados). Além disso, ela fornece a base para o cálculo de diversas métricas de avaliação. Nesta pesquisa, as métricas que serão investigadas são: acurácia, precisão, *recall* e *F1-Score* (KUBAT, 2017).

A acurácia mede a proporção de predições corretas (positivas e negativas) em relação ao total de amostras. Ela é útil em cenários com classes balanceadas, mas pode ser enganosa em contextos desbalanceados, como em detecção de intrusões, onde os ataques podem representar apenas uma pequena fração do tráfego. Nesse caso, um modelo que

classifica todas as amostras como benignas pode atingir alta acurácia, mas falha completamente em detectar ataques. A Eq. 2.1 mostra como é feito o cálculo da acurácia:

$$\text{Acurácia} = \frac{VP + VN}{VP + VN + FP + FN} \tag{2.1}$$

A precisão avalia a proporção de instâncias corretamente classificadas como positivas em relação ao total de instâncias preditas como positivas. Essa métrica é especialmente relevante em cenários em que o custo de um falso positivo é alto. Em outras palavras, quando é mais problemático classificar algo incorretamente como positivo do que deixar de identificá-lo. Assim, a precisão é útil para avaliar a confiabilidade das previsões positivas de um modelo. Entretanto, a precisão sozinha não garante um bom desempenho em detecção de intrusões, já que o modelo pode evitar alarmes falsos deixando de detectar ataques reais. A Eq. 2.2 mostra como é feito o cálculo da precisão:

$$\text{Precisão} = \frac{VP}{VP + FP} \tag{2.2}$$

O *recall*, também chamado de sensibilidade ou revocação, mede a proporção de instâncias positivas corretamente identificadas pelo modelo em relação ao total de instâncias realmente positivas. O *recall* é importante em contextos onde o custo de um falso negativo é elevado, ou seja, quando é crítico não deixar de identificar casos positivos. Assim, o *recall* avalia a capacidade do modelo de capturar todos os casos positivos, mesmo que isso implique maior risco de falsos positivos. Essa métrica é crucial em sistemas de detecção de intrusão, pois reflete a capacidade do modelo de identificar ataques. Um baixo *recall* significa que muitos ataques passam despercebidos (altos FN), o que compromete seriamente a segurança da rede. A Eq. 2.3 mostra como é feito o cálculo do *recall*:

$$\text{Recall} = \frac{VP}{VP + FN} \tag{2.3}$$

Por fim, o *F1-Score* é a média harmônica entre precisão e *recall*, combinando ambas em uma única métrica de avaliação. A principal vantagem do *F1-Score* é equilibrar os dois aspectos: evitar falsos positivos em excesso (alta precisão) e reduzir falsos negativos (alto *recall*). Ele é particularmente útil em cenários com dados desbalanceados ou quando tanto a precisão quanto a sensibilidade são importantes para avaliar o desempenho do modelo. Como a média harmônica penaliza valores muito discrepantes, o *F1-Score* só será alto se tanto a precisão quanto o *recall* forem altos de forma simultânea. A Eq. 2.4 mostra como é feito o cálculo do *F1-Score*:

$$\text{F1-Score} = 2 \cdot \frac{\text{Precisão} \cdot \text{Recall}}{\text{Precisão} + \text{Recall}} \tag{2.4}$$

2.2 Trabalhos Relacionados

A detecção de intrusões em ambientes IoT tem se consolidado como um importante tópico de investigação em cibersegurança. Nesse cenário, métodos de aprendizado de máquina têm sido amplamente utilizados, visando aprimorar a identificação de tráfego malicioso e garantir maior resiliência às redes. Entre os conjuntos de dados que buscam permitir o estudo desses ataques em redes IoT, o *Canadian Institute for Cybersecurity IoT 2023 Dataset* (CICIoT2023) vem recebendo grande atenção. Essa base de dados, ao reunir uma ampla diversidade de ataques e cenários realistas, tem sido amplamente explorada em estudos que buscam avaliar a eficácia de diferentes algoritmos.

Paralelamente, observa-se na literatura um interesse pela estratégia de classificação hierárquica, uma abordagem promissora para problemas de detecção de intrusões em que as classes apresentam relações de dependência ou estrutura taxonômica. Diferentemente da classificação plana, a classificação hierárquica busca organizar os rótulos em níveis ou categorias inter-relacionadas, o que permite explorar melhor a natureza do problema, reduzir a complexidade da decisão em uma única etapa e, potencialmente, aumentar a interpretabilidade e a precisão do modelo.

A análise da literatura é fundamental para compreender quais estratégias já foram exploradas e quais desafios permanecem abertos. Dessa forma, esta seção apresenta e discute os principais trabalhos relacionados ao tema desta monografia, destacando suas contribuições, limitações e potenciais pontos de aprimoramento. Portanto, esta seção será organizada em duas subseções: a primeira abordará investigações baseadas no conjunto de dados CICIoT2023, destacando estratégias de classificação, algoritmos aplicados e resultados alcançados; a segunda discutirá estudos voltados à classificação hierárquica em geral, enfatizando suas contribuições e relevância para o contexto de segurança. O objetivo é estabelecer uma visão crítica do estado da arte que oriente e justifique as escolhas metodológicas adotadas neste trabalho.

2.2.1 Investigações baseadas no CICIoT2023

Os sistemas de detecção de intrusão (IDS) têm ganhado destaque na segurança da informação devido ao aumento das ameaças cibernéticas e à complexidade dos ambientes computacionais, incluindo a Internet das Coisas (IoT). Estudos recentes têm avaliado diferentes abordagens, como o de Maseer et al. (2021), que realizou um *benchmarking* de algoritmos supervisionados e não supervisionados no conjunto de dados CICIDS2017, destacando o bom desempenho de *K-Nearest Neighbors* (KNN), *Decision Tree* e *Naive Bayes*. No contexto da IoT, (HASAN et al., 2019) comparou modelos de aprendizado de máquina usando um conjunto de tráfego IoT aberto, apontando *Decision Tree*, *Random Forest* e *Artificial Neural Networks* como melhores, com destaque para o *Random Forest*

pela acurácia na classificação de múltiplos tipos de ataque.

O trabalho de Neto et al. (2023) propõe um novo e extenso conjunto de dados de ataques IoT, chamado CICIoT2023, com o objetivo de simular cenários realistas de tráfego de rede em ambientes IoT e possibilitar a avaliação de sistemas de detecção de intrusões baseados em técnicas de aprendizado de máquina. Os autores executaram 33 ataques distintos numa topologia IoT, classificando-os em sete categorias principais (*DDoS*, *DoS*, *Recon*, *Web-Based*, *Brute Force*, *Spoofing* e *Mirai*), sendo que todos os ataques foram realizados por dispositivos IoT maliciosos visando outros dispositivos IoT. Os dados de tráfego de rede benigno e malicioso foram coletados e uma avaliação de aprendizado de máquina foi feita para demonstrar a viabilidade do conjunto CICIoT2023 para treinar modelos de detecção e classificação de ataques IoT.

Para isso, foram avaliados cinco algoritmos de ML (Regressão Logística, *Perceptron*, *Adaboost*, *Random Forest* e *Deep Neural Network*) em três cenários de classificação: binária (benigno/malicioso), multiclasse em oito classes (sete categorias de ataque mais benigno) e multiclasse em 34 classes (33 ataques individuais mais benigno). Os resultados indicaram que, embora todos os métodos tenham apresentado alta performance na classificação binária, houve uma degradação perceptível quando a tarefa envolveu oito classes. Ainda assim, o *Random Forest* e a *Deep Neural Network* se destacaram por manterem um desempenho superior em comparação aos demais modelos nesse cenário. A classificação multiclasse em 34 classes revelou-se a mais desafiadora, com o *Random Forest* e a *Deep Neural Network* apresentando os melhores resultados, apesar de uma ligeira redução no desempenho. As matrizes de confusão indicaram que classes com muitos exemplos, como *DDoS*, *DoS* e *Mirai*, apresentam baixas taxas de erro, seguidas por *Recon* e *spoofing*. No entanto, ataques web e de *brute force* apresentam confusões frequentes, sendo identificados como benignos, *Recon* ou *spoofing*.

A literatura recente tem apresentado um número crescente de estudos voltados à detecção de ciberataques em ambientes IoT, muitos dos quais utilizam o conjunto de dados CICIoT2023 como referência para experimentação e avaliação de modelos. Kumar, Rastogi e Ranga (2024), por exemplo, avalia o desempenho de diversos classificadores de aprendizado de máquina treinados e testados nessa base. A pesquisa empregou algoritmos como *Support Vector Machine*, Regressão Logística, *Decision Tree*, *AdaBoost* e *Random Forest*. A abordagem metodológica incluiu a classificação dos ataques em três cenários distintos: com 34 classes, 8 classes e 2 classes, permitindo a análise do impacto da granularidade na performance. Os autores compararam métricas de desempenho como Acurácia, *Recall*, Precisão e *F1-Score* para cada algoritmo. Segundo os autores, os resultados do estudo mostraram que a performance dos modelos de aprendizado de máquina aumenta consistentemente à medida que o número de classes é reduzido, indicando que uma abordagem de classificação mais focada e menos granular beneficia a acurácia dos

modelos. Embora o desempenho geral sofra com mais categorias de ataque, algoritmos como *Random Forest* e *Decision Tree* demonstraram alta eficácia mesmo em cenários de alta granularidade, mantendo acurácia e *F1-score* elevados. O *Support Vector Machine*, por outro lado, teve o pior desempenho em todos os cenários, especialmente nos de maior granularidade, o que sugere suas limitações para tarefas de classificação mais detalhadas de ciberataques.

De maneira semelhante, Hafezian, Naderan e Jaderyan (2024) investiga uma abordagem baseada em aprendizado de máquina para detecção e classificação de intrusões multi-classe em ambientes IoT, usando o CICIoT2023. A metodologia adotada envolveu o treinamento e a avaliação de seis algoritmos supervisionados: Regressão Logística, *AdaBoost*, *Perceptron*, *Multi-Layer Perceptron* (MLP), *Random Forest* e *Hist-Gradient Boosting*. Esses modelos foram aplicados em três modos de classificação distintos: binário, oito classes e 34 classes. Os resultados da avaliação indicaram que o algoritmo *Random Forest* alcançou os melhores resultados de classificação em todos os três modos e para todas as métricas de avaliação, como acurácia, precisão, *recall* e *F1-score*. Além disso, o algoritmo *Hist-Gradient Boosting* demonstrou o melhor desempenho em termos de tempo de treinamento. Assim como na pesquisa de Kumar, Rastogi e Ranga (2024), os resultados da avaliação mostraram que as métricas de desempenho diminuíram com o aumento da complexidade do problema e, embora os algoritmos ainda apresentassem alta acurácia, a capacidade de identificar e classificar ataques específicos com maior precisão e *recall* foi afetada negativamente quanto maior o número de classes a serem distinguidas.

A monografia de Esteves (2025) aborda a aplicação de Inteligência Artificial Explicável (XAI) em Sistemas de Detecção de Intrusão (IDSs) para dispositivos IoT, visando aumentar a transparência e a confiança nessas soluções. A metodologia envolveu o pré-processamento e análise exploratória do conjunto de dados CICIoT2023, seguido pelo treinamento de oito modelos de aprendizado de máquina, com o *XGBoost* sendo selecionado pelo seu desempenho. A pesquisa focou em dois cenários de classificação com a aplicação da técnica SHAP (*SHapley Additive exPlanations*) para gerar explicações globais e locais. No cenário de classificação binária, que diferencia tráfego normal de tráfego de ataque, o *XGBoost* alcançou 86% de acurácia. Já no cenário multiclasse, onde o objetivo era separar múltiplos tipos de ataques, o modelo obteve 77% de acurácia, apresentando ressalvas em categorias minoritárias e uma revocação de aproximadamente 38% nos melhores casos.

Embora o CICIoT2023 seja um recurso valioso e amplamente adotado, os estudos atuais deixam em aberto importantes oportunidades de pesquisa, sobretudo no que diz respeito à exploração de abordagens hierárquicas e à análise aprofundada das limitações dos classificadores multiclasse. Primeiramente, nota-se que todos os trabalhos apresentam uma queda de desempenho quando o problema é formulado em múltiplas classes, em comparação com a detecção binária entre tráfego benigno e malicioso. Apesar disso, nenhum

estudo identificado propõe estratégias intermediárias que conciliem os pontos fortes de cada abordagem - a filtragem eficiente de ataques no cenário binário e a capacidade de distinção entre múltiplos tipos de ataque no cenário multiclasse. Além disso, nenhum dos trabalhos explora a classificação hierárquica como alternativa metodológica. Essa lacuna é particularmente relevante, uma vez que tal estratégia poderia oferecer um equilíbrio entre as qualidades dos dois classificadores. Outro aspecto crítico é a ausência, na maior parte das investigações, de análises mais detalhadas sobre os erros de classificação. Essa limitação dificulta a identificação de classes problemáticas e a compreensão de quais tipos específicos de ataques ainda representam desafios significativos para os modelos.

2.2.2 Investigações baseadas na Classificação Hierárquica

Ao longo das últimas duas décadas, diversos trabalhos investigaram abordagens hierárquicas com o objetivo de aprimorar a qualidade das decisões dos classificadores. A pesquisa de Lorena, Carvalho e Gama (2008) apresenta uma revisão abrangente sobre a combinação de classificadores binários em problemas de classificação multiclasse, focando na decomposição do problema original em múltiplos subproblemas binários. Essa estratégia é motivada pela redução da complexidade e pela adaptabilidade de algoritmos que não são facilmente reformulados para contextos multiclasse. O trabalho destaca estratégias hierárquicas de decomposição, que organizam classificadores binários em estruturas como árvores binárias direcionadas ou Grafos Acíclicos Direcionados (DAGs). Nelas, a classificação ocorre em etapas: discriminações mais gerais são realizadas inicialmente e progressivamente refinadas até a predição final. Cada nó interno da hierarquia corresponde a um classificador binário que distingue dois subconjuntos de classes, enquanto os nós terminais representam as classes individuais. O processo de classificação de uma nova instância é iterativo: inicia-se no nó raiz e, com base na decisão de cada classificador binário, um novo nó é visitado até que um nó folha seja alcançado, determinando a classe final. O trabalho Silla Jr e Freitas (2011) faz uma revisão bibliográfica sobre a tarefa de classificação hierárquica. Os autores clarificam o que é classificação hierárquica, distinguindo-a da classificação plana, que ignora completamente essa estrutura de classes e é tratada como um problema multi-classe padrão. A pesquisa propõe uma nova taxonomia para classificar problemas e algoritmos de classificação hierárquica, organizando as abordagens existentes em planas, locais e globais. É realizada uma revisão crítica e comparação empírica dessas abordagens, destacando suas vantagens e desvantagens. Embora os artigos apresentem uma revisão conceitual das abordagens hierárquicas, os princípios e métodos descritos são diretamente aplicáveis e fornecem uma base teórica sólida para sistemas como a classificação hierárquica empregada na detecção de intrusões, onde a decomposição de problemas multiclasse em problemas de menor complexidade é crucial para um desempenho eficaz.

Nesse contexto, Sarıkaya e Kılıç (2020) propõe um modelo hierárquico multiclasse para detecção de intrusões, visando melhorar as taxas de detecção de classes de ataque que são historicamente difíceis de identificar, usando o conjunto de dados UNSW-NB15. A abordagem hierárquica emprega classificadores *Random Forest* em múltiplas etapas, transformando tarefas de classificação multiclasse em tarefas de classificação binária para as classes de baixo desempenho. A estrutura do modelo é construída em estágios, onde cada classificador tem um propósito específico de detecção de ataque. Primeiramente, no Estágio-1, um classificador *Random Forest* é treinado para detectar se o tráfego de rede é um ataque ou normal. Se um ataque for detectado, o processo avança para o Estágio-2, onde um segundo classificador *Random Forest* determina se o ataque pertence ao Grupo 1 (*DoS*, *Exploit*) ou ao Grupo 2 (Outras Classes de Ataque), o que é crucial para separar as classes com baixas taxas de detecção. Finalmente, no Estágio-3, existem dois classificadores *Random Forest* distintos: um para detectar especificamente *DoS*/*Exploit* se a instância for classificada no Grupo 1, e outro para classificar as demais classes de ataque (Análise, *Backdoor*, *Fuzzers*, Reconhecimento, Genérico, *Shellcode*, *Worms*) se a instância for classificada no Grupo 2. Este modelo alcançou uma acurácia geral de classificação de 80,78%, superando o classificador *Random Forest* de linha de base e significativamente aumentando as taxas de detecção para os ataques *DoS*, *Exploit* e *Fuzzers*, de acordo com os autores.

O trabalho Alin et al. (2020) propõe uma nova metodologia hierárquica baseada em algoritmos de aprendizado profundo para detecção de intrusões, visando superar as limitações dos métodos tradicionais de proteção que são frequentemente ineficazes e caros em recursos. A abordagem central do estudo consiste em uma estratégia de classificação em dois níveis: inicialmente, realiza-se uma classificação binária (Normal/Ataque) para detectar rapidamente conexões anormais, alertando o administrador; em seguida, para as conexões identificadas como 'ataque', aplica-se uma classificação multiclasse para detalhar o tipo específico de intrusão. A comparação com abordagens de classificação multiclasse tradicionais revelou que a metodologia hierárquica proposta demonstrou um desempenho consideravelmente superior nos conjuntos KDD-CUP99, NSL-KDD, UNSW-NB15 e CIC-IDS2017, alcançando uma taxa de detecção de intrusões de até 95%.

O sistema de detecção de intrusões proposto por Sarnovsky e Paralic (2020) emprega uma abordagem de predição hierárquica em múltiplas etapas, que integra modelos de aprendizado de máquina com uma ontologia para otimizar a detecção de ataques de rede. O processo inicia-se com uma primeira fase de classificação binária onde um classificador, baseado em árvores de decisão, distingue o tráfego normal de conexões de ataque. Se uma conexão é identificada como ataque, ela avança para a segunda fase, guiada pela ontologia, que navega pela taxonomia de ataques (*DoS*, *R2L*, *U2R*, *Probe*). Nesta fase, um *ensemble model* com votação ponderada é utilizado para prever a classe do ataque. Este modelo é composto por classificadores base de *Naive Bayes* e Árvores de Decisão.

Os modelos baseados em árvores desempenhavam bem na previsão de ataques *Probe*, *DoS* e *R2L*, mas falhavam na detecção de *U2R*. Para superar isso, modelos separados "*one-vs-all*"foram treinados especificamente para detectar apenas os ataques *U2R*. Assim, o ensemble aproveita os modelos baseados em árvores para as classes majoritárias e os modelos "*one-vs-all*"para *U2R*, combinando suas predições através de um esquema de votação ponderada. Subsequentemente, modelos específicos são acionados para prever o tipo concreto de ataque dentro da classe identificada. O sistema foi avaliado no conjunto de dados KDD-99, demonstrando um desempenho superior em comparação com métodos similares, segundo os autores.

O trabalho de (MOHD; SINGH; BHADAURIA, 2021) propõe um Sistema de Detecção de Intrusões (IDS) baseado em classificadores hierárquicos híbridos para enfrentar o desafio da segurança em sistemas de rede. Para isso, foi utilizado o banco de dados KDD-99 atualizado, que consiste em 4.898.431 instâncias de tráfego de rede, incluindo classes normais e quatro tipos de ataques: *DoS*, *U2R*, *R2L* e *Probing*. A metodologia inicial envolveu o uso de seis classificadores de aprendizado de máquina - *Support Vector Machine (SVM), Probabilistic Neural Network (PNN), Decision Tree (DT), Neuro-fuzzy Classifier (NFC), Smooth Support Vector Machine (SSVM) e k-Nearest Neighbor (kNN)* - cada um aplicado em uma estrutura de classificação hierárquica de quatro níveis. Cada nível tinha uma função específica: o Nível-1 separava tráfego normal de ataque; o Nível-2 identificava ataques *DoS*; o Nível-3 classificava ataques *R2L*; e o Nível-4 diferenciava ataques *U2R* e de sondagem (*probing*). Para construir o modelo hierárquico híbrido, os pesquisadores identificaram os classificadores com o melhor desempenho em cada nível individual. Assim, o SSVM-1 foi escolhido para o Nível-1, o NFC-2 para o Nível-2, o SVM-3 para o Nível-3 e o kNN-4 para o Nível-4. A combinação desses melhores classificadores resultou no sistema híbrido proposto, que alcançou uma Taxa Geral de Detecção (ODA) de 98,79%.

No contexto de IoT, o trabalho (UDDIN et al., 2025) investiga a eficácia da classificação hierárquica em Sistemas de Detecção de Intrusões (IDS) com uma ênfase particular na proteção contra ameaças cibernéticas em redes de tecnologia emergentes, incluindo a Internet das Coisas. Os autores propõem um modelo de três níveis para abordar a estrutura inerente dos ciberataques. A abordagem envolve a classificação de tráfego em três estágios: primeiro, distinguindo entre tráfego benigno e ataque (Nível 1); segundo, categorizando tipos de ataque de grão grosso (famílias de ataque) (Nível 2); e terceiro, identificando subtipos de ataque específicos e de grão fino (Nível 3). Nesse estudo, 10 diferentes classificadores de aprendizado de máquina foram avaliados em 10 conjuntos de dados IDS contemporâneos, incluindo bases específicas para IoT, como ToN-IoT-Network, ToN-IoT-IoTs, BoT-IoT e XIIOTID. Embora o desempenho geral (precisão, *recall* e *F1-score*) seja comparável ao da classificação plana, a abordagem hierárquica minimiza significativamente a classificação incorreta de ataques como tráfego normal (falsos negativos), o que é

crucial para a segurança cibernética. Segundo os autores, a base de dados ToN-IoT-IoTs apresentou um desempenho relativamente inferior na detecção de ataques em cenários de IoT, sugerindo a necessidade de melhorias adicionais na detecção desse tipo de ataque.

Embora a classificação hierárquica já tenha sido aplicada em sistemas de detecção de intrusão, sua utilização no contexto específico de redes IoT ainda é pouco explorada. Essa lacuna abre um campo promissor para novas investigações que considerem simultaneamente os desafios inerentes ao tráfego IoT e o potencial da classificação hierárquica em estruturar as classes de forma organizada, possibilitando ganhos tanto em desempenho quanto em interpretabilidade. O trabalho (UDDIN et al., 2025) chega a investigar a aplicação dessa abordagem para a detecção de intrusões em redes IoT. Contudo, sua proposta não aprofunda a exploração das características diferenciais dos ataques ao longo da hierarquia, nem investiga como essa estrutura pode contribuir para melhorar a detecção de cada classe de ataque. Dessa forma, esta monografia se diferencia ao propor uma investigação que integra a classificação hierárquica ao domínio de detecção de intrusões em IoT, com o objetivo de explorar não apenas a viabilidade dessa abordagem, mas também seu impacto na melhoria da robustez e da precisão na identificação de ataques, superando as limitações observadas nos estudos anteriores.

3 Desenvolvimento

Neste capítulo, será apresentada a metodologia adotada para o desenvolvimento deste trabalho. Serão descritas as etapas seguidas, as decisões de projeto e os critérios que orientaram a implementação da solução proposta, de forma a permitir o entendimento claro do processo de construção desta pesquisa. Inicialmente, é apresentada uma visão geral do processo metodológico, seguida do aprofundamento em cada um desses passos.

3.1 Visão Geral

Este trabalho investiga uma abordagem de classificação hierárquica com o objetivo de aprimorar a detecção de ataques em redes IoT. Conforme apresentado na Figura 2, o desenvolvimento dos modelos de aprendizado de máquina seguiu quatro etapas principais: (i) extração do conjunto de dados, (ii) pré-processamento e transformação dos dados, (iii) construção dos modelos e (iv) avaliação dos modelos.

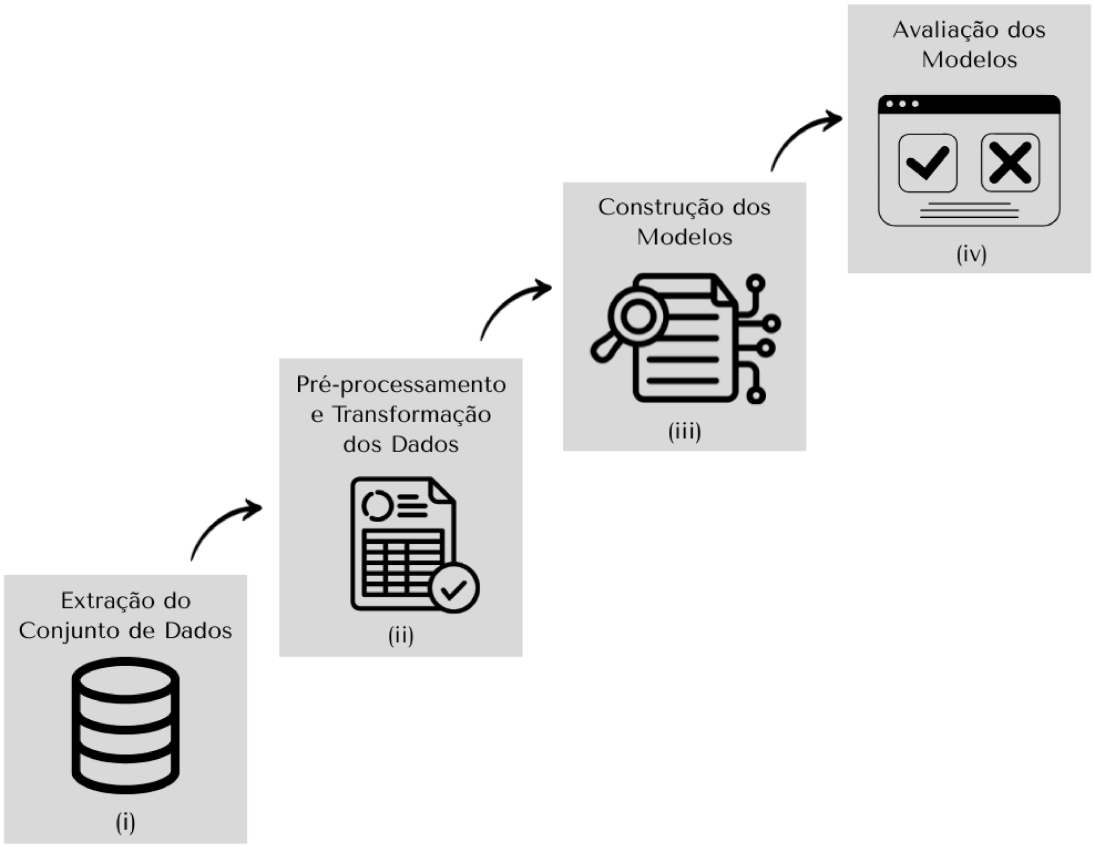

Figura 2 – Metodologia de desenvolvimento deste trabalho.

A primeira etapa, de seleção de dados, consiste na análise e escolha de um con-

junto de dados adequado, que seja relevante e representativo para o problema de ataques direcionados a dispositivos e redes IoT. A etapa seguinte, de pré-processamento e transformação, tem como finalidade preparar os dados para o treinamento dos modelos e pode ser subdividida em duas fases: tratamento e transformação. O tratamento envolve procedimentos como limpeza de inconsistências, remoção de duplicatas e tratamento de valores ausentes, garantindo maior qualidade ao conjunto de dados. Já a transformação refere-se à aplicação de técnicas que tornam os dados mais adequados ao aprendizado, incluindo normalização ou padronização, codificação de variáveis categóricas e, quando necessário, a seleção ou extração de características. A terceira etapa corresponde à construção dos modelos de aprendizado de máquina, cujo objetivo é identificar padrões no tráfego de rede capazes de distinguir entre eventos normais e ataques. Por fim, a avaliação dos modelos é realizada por meio da aplicação de métricas selecionadas, permitindo analisar os desempenhos obtidos e medir o quanto as soluções propostas contribuem para a detecção de intrusões em ambientes IoT.

Todo o código desenvolvido neste trabalho, incluindo a sequência de etapas completa de extração, tratamento e transformação dos dados, além da construção e avaliação dos classificadores planos e hierárquicos, está disponível publicamente no repositório GitHub, acessível em:

<https://github.com/gioomartins/Classificacao-Hierarquica-IDS-CICIoT2023.git>

O acesso ao código permite reproduzir os experimentos apresentados e explorar os modelos implementados, garantindo transparência e reprodutibilidade dos resultados.

3.2 Extração do Conjunto de Dados

A produção de dados de segurança para a Internet das Coisas que possam apoiar aplicações reais é desafiadora, principalmente pela dificuldade de simular uma rede extensa com diversos dispositivos IoT, semelhante às topologias de aplicações IoT no mundo real. Nesse contexto, (NETO et al., 2023) propõe um novo e extenso conjunto de dados de ataques IoT, denominado CICIoT2023, com o objetivo de impulsionar o desenvolvimento de soluções de análise de segurança aplicáveis a operações reais em ambientes IoT. Este conjunto de dados foi utilizado neste trabalho para o treinamento e a avaliação dos modelos propostos.

Para simular cenários realistas de tráfego em redes IoT, os autores executaram 33 ataques distintos em uma topologia composta por 105 dispositivos reais. O conjunto de dados resultante contém aproximadamente 46,6 milhões de registros rotulados (NARAYAN et al., 2023), sendo cada registro uma instância de tráfego de rede descrita por 46 atributos, que contemplam métricas de pacotes e fluxos, como taxa de pacotes, dura-

Tabela 2 – Número de registros para cada ataque e categoria.

Classe	Tipo de Ataque	Qtd. de Registros	Total por Classe (CICIoT2023)	Total por Classe (Subamostragem)
DDoS	DDoS-ICMP_Flood	7.200.504	33.984.560	10.540.341
	DDoS-UDP_Flood	5.412.287		
	DDoS-TCP_Flood	4.497.667		
	DDoS-PSHACK_Flood	4.094.755		
	DDoS-SYN_Flood	4.059.190		
	DDoS-RSTFINFlood	4.045.285		
	DDoS-SynonymousIP_Flood	3.598.138		
	DDoS-UDP_Fragmentation	286.925		
	DDoS-ACK_Fragmentation	285.104		
	DDoS-ICMP_Fragmentation	452.489		
	DDoS-HTTP_Flood	28.790		
	DDoS-SlowLoris	23.426		
DoS	DoS-UDP_Flood	3.318.595	8.090.738	3.863.993
	DoS-HTTP_Flood	71.864		
	DoS-TCP_Flood	2.671.445		
	DoS-SYN_Flood	2.028.834		
Mirai	Mirai-greeth_flood	991.866	2.634.124	1.268.509
	Mirai-udpplain	890.576		
	Mirai-greip_flood	751.682		
Benign	BenignTraffic	1.098.195	1.098.195	658.906
Spoofing	DNS_Spoofing	178.911	486.504	291.878
	MITM-ArpSpoofing	307.593		
Recon	Recon-PingSweep	2.262	354.565	212.534
	Recon-HostDiscovery	134.378		
	Recon-OSScan	98.259		
	Recon-PortScan	82.284		
	VulnerabilityScan	37.382		
Web	Uploading_Attack	1.252	24.829	14.897
	BrowserHijacking	5.859		
	CommandInjection	5.409		
	SqlInjection	5.245		
	XSS	3.846		
	Backdoor_Malware	3.218		
BruteForce	DictionaryBruteForce	13.064	13.064	7.839

ção, *flags* TCP, entre outros. Em termos de rotulagem, os 33 tipos distintos de ataques podem ser organizados em sete categorias de ataques principais, além do tráfego benigno, conforme a Tabela 2. Portanto, as classes investigadas no conjunto de dados resultante são compostas por: *Benign*, *DDoS*, *DoS*, *Recon*, *Web-Based*, *Brute Force*, *Spoofing* e *Mirai* (NETO et al., 2023).

Um aspecto relevante é que o conjunto apresenta alto desbalanceamento de classes, com grande disparidade no número de instâncias entre as categorias. O desbalanceamento é um desafio recorrente em detecção de intrusões, pois tende a enviesar os modelos de aprendizado de máquina em favor das classes majoritárias, comprometendo a detecção de ataques menos frequentes, porém críticos. Modelos treinados nesse contexto podem alcançar valores elevados de acurácia global, mas falham na correta identificação de intrusões críticas (SHANMUGAM; RAZAVI-FAR; HALLAJI, 2025).

Considerando o grande volume de instâncias do CICIoT2023, este trabalho em-

pregou uma subamostra estratificada correspondente a 60% do conjunto original. A estratificação foi necessária para preservar a proporção entre as classes, aspecto essencial em cenários de desbalanceamento, garantindo que ataques com menor representação relativa fossem mantidos no processo de treinamento e teste. Os quantitativos das categorias da subamostragem também são apresentados na Tabela 2.

3.3 Pré-processamentos e Transformação dos Dados

A preparação do conjunto de dados CICIoT2023 para a etapa de modelagem foi organizada em duas fases complementares: tratamento e transformação. Esse processo garante tanto a integridade quanto a adequação dos dados, possibilitando a aplicação eficiente das técnicas de aprendizado de máquina.

O tratamento foi voltado para assegurar a consistência e a qualidade do conjunto antes da construção dos modelos. A presença de valores ausentes (*NaN*) representa um problema crítico, pois a maioria dos algoritmos de aprendizado de máquina não é capaz de lidar nativamente com esse tipo de informação. Além disso, valores ausentes em atributos relevantes comprometem a extração de padrões, reduzindo a capacidade de generalização do modelo. Outro aspecto tratado foi a remoção de registros duplicados, uma vez que a repetição de amostras pode super-representar determinados padrões, fazendo com que o modelo aprenda a reconhecer com maior facilidade esses casos em detrimento de outros menos frequentes. Esse fenômeno, além de enviesar o processo de aprendizado, pode inflar artificialmente métricas de avaliação como acurácia e *recall*, transmitindo uma percepção equivocada do desempenho do modelo. Para lidar com essas questões, foram aplicadas as funções *dropna()* e *drop_duplicates()* da biblioteca *pandas* (Pandas Developers, 2025), para remover registros com valores ausentes e duplicatas, assegurando maior integridade do conjunto e robustez no processo de modelagem.

Na etapa de transformação, foram aplicadas técnicas destinadas a tornar os dados mais apropriados para o aprendizado. Neste trabalho, foram considerados todos os 46 atributos originais do conjunto de dados CICIoT2023. Quanto ao atributo alvo (*label*), que contém os rótulos de tráfego benigno e de ataques, os 33 tipos de ataques originais foram agrupados em sete categorias principais: *DDoS*, *DoS*, *Mirai*, *Recon*, *Spoofing*, *Web-Based* e *Brute Force*, além da classe *Benign*. Esse mapeamento teve como objetivo reduzir a complexidade do problema de classificação e alinhar as análises aos objetivos deste trabalho. Por fim, foi empregada a técnica de padronização dos atributos contínuos por meio da função *StandardScaler()* da biblioteca *scikit-learn* (Scikit-learn Developers, 2025). Esse procedimento foi aplicado ao conjunto completo de dados, ajustando os atributos para média zero e variância unitária, de modo a garantir que variáveis com escalas distintas não exercessem influência desproporcional no processo de aprendizado.

Adicionalmente, devido ao elevado volume do conjunto CICIoT2023, com milhões de registros e dezenas de atributos, uma etapa essencial deste trabalho consistiu na otimização do uso de memória. Esse procedimento visou reduzir o consumo de recursos computacionais durante o processamento, sem perda de informação, tornando o treinamento e a avaliação dos modelos mais eficientes. Inspirada no trabalho de Esteves (2025), a estratégia implementada consistiu em uma rotina para sugestão e aplicação de tipos de dados mais econômicos a cada atributo do conjunto. Esse processo avaliou o intervalo de valores de cada coluna e, com base nesse diagnóstico, mapeou os tipos de dados originais para representações mais compactas. Por exemplo, atributos inteiros foram convertidos para tipos como int8, int16, int32 ou int64, dependendo dos valores mínimo e máximo encontrados. Após essa otimização, foi feita a comparação entre o consumo de memória original e o consumo após a transformação, permitindo quantificar a economia obtida, como pode ser visto na Tabela 3. O procedimento reporta tanto a redução absoluta em megabytes quanto a economia percentual, demonstrando o ganho de eficiência alcançado.

Tabela 3 – Otimização de memória.

Memória Original	**Memória Otimizada**	**Memória Economizada**
19253,31 MB	4764,05 MB	14489,26 MB (75%)

3.4 Construção dos Modelos

Após a etapa de transformação, o conjunto de dados foi particionado em dois subconjuntos: treino e teste. O primeiro foi utilizado no processo de construção dos modelos, enquanto o segundo serviu exclusivamente para a avaliação de desempenho.

O processo de treinamento é a fase em que o algoritmo de aprendizado de máquina analisa os exemplos de treino a fim de identificar padrões e relações entre os atributos, de modo a construir um modelo capaz de realizar classificações em amostras não vistas. A Figura 3 ilustra o treinamento dos modelos desenvolvidos neste trabalho.

Nesta monografia, foram propostos e avaliados quatro modelos distintos para o problema de detecção de intrusões em ambientes IoT. Os modelos Binário e Multiclasse adotam a abordagem de classificação plana, sendo implementados com o objetivo de estabelecer uma linha de base para comparação. Já os modelos BDM e BRDM, explicados nas seções 3.4.3 e 3.4.4, foram desenvolvidos segundo a estratégia de classificação hierárquica, constituindo o foco central desta pesquisa.

3.4.1 Modelo Binário

O modelo binário tradicional foi desenvolvido com o objetivo de distinguir entre tráfego benigno e tráfego malicioso, independentemente da categoria específica de ataque.

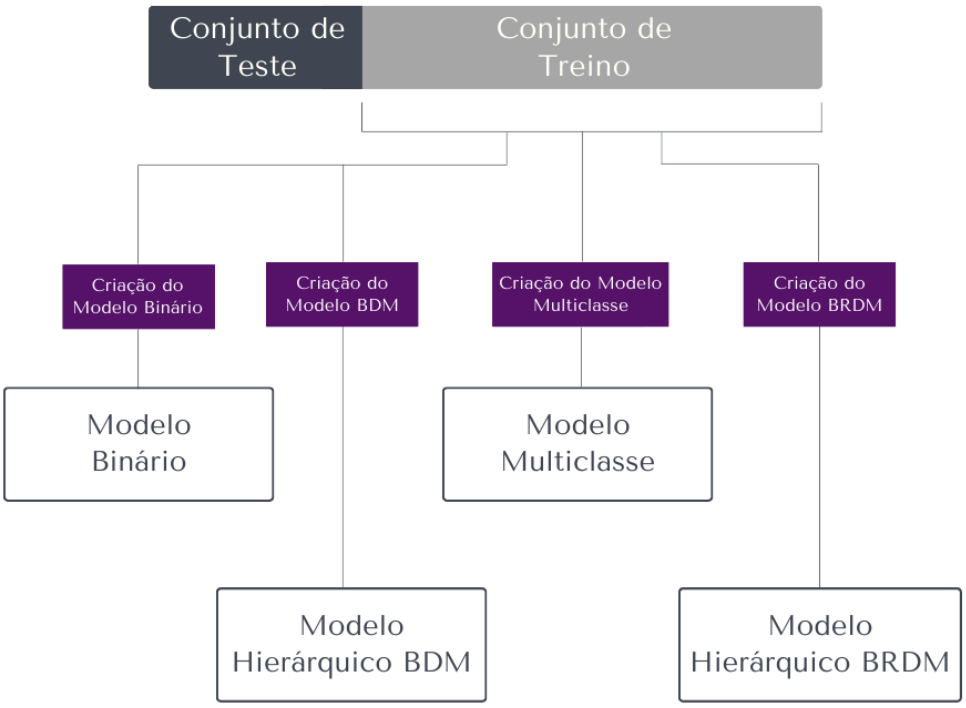

Figura 3 – Fluxo de treinamento dos modelos desenvolvidos neste trabalho.

Para isso, o atributo alvo (*label*) do conjunto de treino foi transformado em uma variável binária, em que todas as amostras originalmente rotuladas como um tipo de ataque foram agrupadas em uma única classe denominada *Malicious* (representada por 1), enquanto as amostras normais permaneceram classificadas como *Benign* (representada por 0).

Essa simplificação do problema permite avaliar a capacidade do modelo em identificar, de forma geral, a presença de atividades anômalas no tráfego de rede, sem considerar inicialmente a diferenciação entre os diversos tipos de ataques. Embora tal abordagem reduza significativamente a complexidade da tarefa de classificação, ela também implica a perda de informações importantes sobre a natureza específica do ataque detectado.

A Figura 4 ilustra, de forma esquemática, as possíveis saídas do modelo binário ao classificar uma determinada amostra.

3.4.2 Modelos Multiclasse

O modelo multiclasse tradicional foi desenvolvido com o objetivo de distinguir entre diferentes categorias de tráfego, incluindo os tipos específicos de ataque. Dessa forma, sua classificação abrange tanto a classe *Benign* quanto sete classes distintas de ataques *DDoS*, *DoS*, *Mirai*, *Recon*, *Spoofing*, *Web-Based* e *Brute Force*. Para tal, o processo de

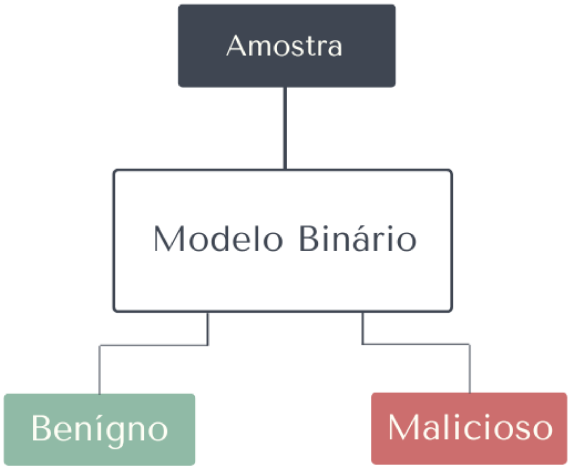

Figura 4 – Fluxo de execução do modelo binário.

aprendizado do modelo considerou o mapeamento original do atributo alvo do conjunto de treino, contendo essas 8 categorias.

Nesse contexto, o modelo foi construído visando identificar não apenas a presença de tráfego malicioso, mas também o tipo específico de ataque em execução. Essa formulação aumenta a granularidade da detecção em relação ao modelo binário, oferecendo maior riqueza informacional para análise e resposta a incidentes.

Entretanto, o problema multiclasse introduz desafios adicionais, uma vez que o modelo deve distinguir entre ataques que frequentemente apresentam padrões semelhantes de tráfego, o que pode levar a maior sobreposição entre classes e, consequentemente, a uma diminuição na detecção de ataques. Ainda assim, essa abordagem é fundamental para avaliar a viabilidade de sistemas de detecção capazes de fornecer diagnósticos mais detalhados sobre a natureza da ameaça.

A Figura 5 apresenta, de forma esquemática, as possíveis saídas do modelo multiclasse para a classificação de uma amostra.

3.4.3 Modelo Hierárquico BDM

O modelo hierárquico BDM (**B**enigno, **D**DoS e **M**ulti-Ataque) envolveu uma abordagem estruturada de três níveis para a classificação do tráfego nas 8 classes originalmente definidas para o conjunto de dados: *Benign*, *DDoS*, *DoS*, *Mirai*, *Recon*, *Spoofing*, *Web-Based* e *Brute Force*. A motivação por trás desse modelo é manter a filtragem eficiente de tráfego malicioso do modelo binário e adicionar a granularidade e riqueza informacional do modelo multiclasse, explorando a possibilidade de hierarquização dos ciberataques, algo que as abordagens de classificação plana não exploram.

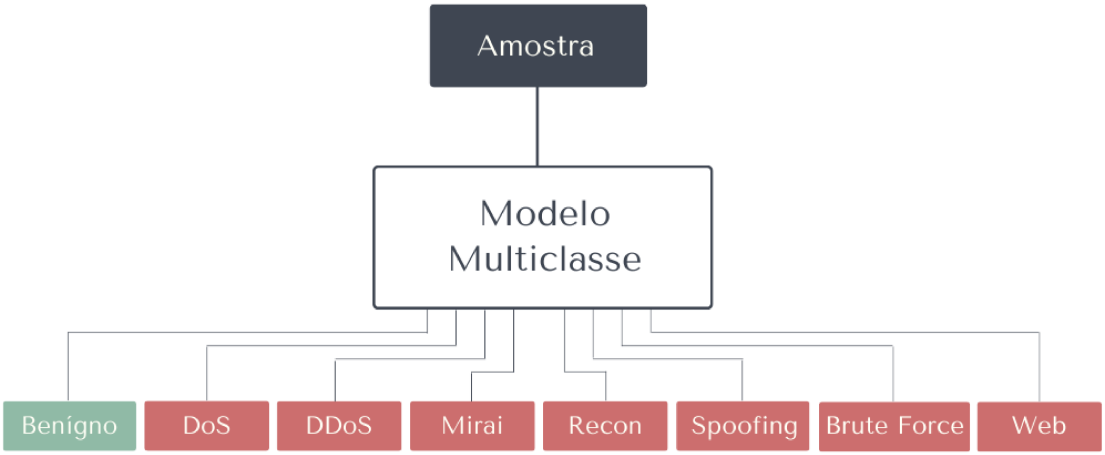

Figura 5 – Fluxo de execução do modelo multiclasse.

O modelo foi projetado com três níveis distintos de classificação, cada um composto por um modelo de classificação isolado. O nome BDM, vem justamente da junção da primeira letra de cada um desses modelos que são: Modelo Benigno, Modelo DDoS e Modelo Multi-Ataque. A Figura 6 ilustra o processo de treinamento de cada um desses modelos.

O Modelo Benigno corresponde a um classificador binário treinado para distinguir, inicialmente, entre tráfego benigno e malicioso. O processo de aprendizado desse classificador foi realizado da mesma forma que no modelo binário tradicional, ou seja, o rótulo das amostras de treino foi mapeado em duas categorias: Benigno e Malicioso.

Em seguida, o Modelo *DDoS* foi treinado. Ele corresponde a um classificador binário treinado para identificar ataques do tipo *DDoS*, classificando as amostras em duas categorias: *DDoS* e Outros Ataques. Para viabilizar o processo de treinamento desse classificador, o conjunto de treinamento original precisou passar por adaptações. Inicialmente, as amostras benignas foram removidas do conjunto, visto que esse tipo de classe é tratado pelo Modelo Benigno. Em seguida, o rótulo das amostras restantes foi mapeado de forma que todos os ataques diferentes de *DDoS* foram agrupados em uma única classe denominada Outros Ataques (representada por 1), enquanto os registros de ataques do tipo *DDoS* permaneceram classificados como *DDoS* (representada por 0).

Por fim, o Modelo Multi-Ataque trata-se de um classificador multiclasse treinado para categorizar as amostras remanescentes em suas respectivas classes, ou seja, identificar qual dos tipos de ataque (sem ser o *DDoS*) foi realizado. Para permitir o processo de aprendizado desse classificador, as amostras de tráfego benigno e de ataques *DDoS* foram previamente removidas do conjunto de treino, uma vez que já são tratadas pelos modelos anteriores. Em seguida, o rótulo das amostras restantes foi mantido de forma explícita,

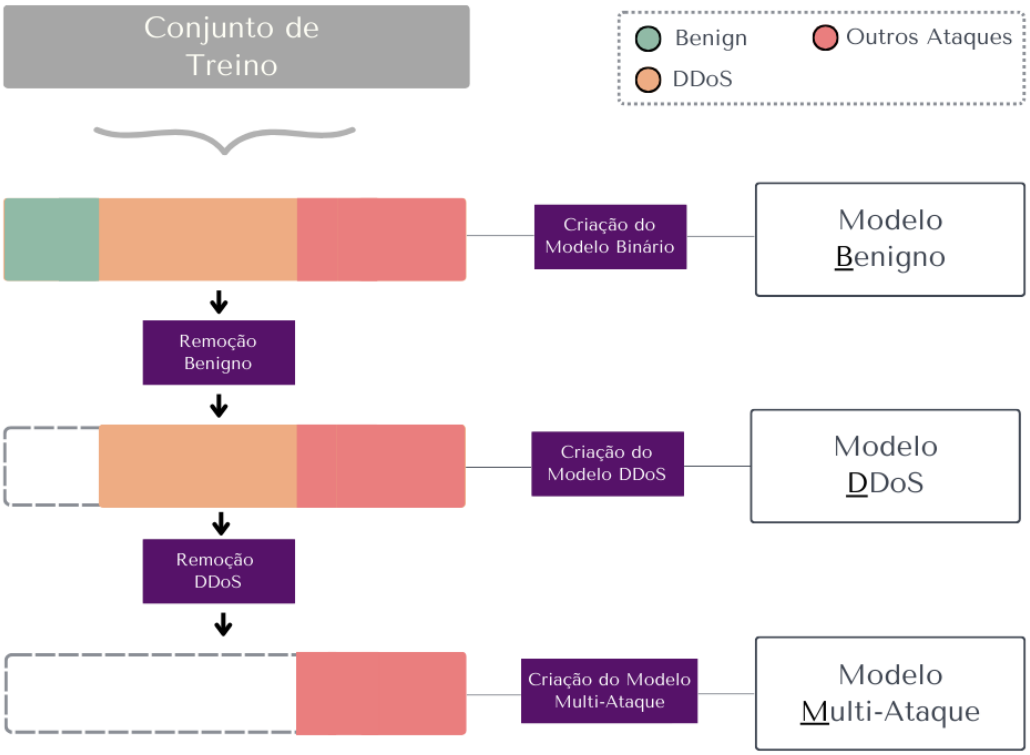

Figura 6 – Fluxo de treinamento dos modelos que compõem o classificador BDM.

preservando a distinção entre os seis tipos de ataques ainda não classificados: *DoS*, *Mirai*, *Recon*, *Spoofing*, *Web-Based* e *Brute Force*. Esse processo é descrito no Algoritmo 1.

O funcionamento básico do classificador BDM, ilustrado na Figura 7, consiste na execução desses 3 modelos previamente treinados.

O nível 1, como apresentado pela figura, serve como uma primeira camada de filtragem, que aproveita a eficiência do modelo binário em separar o tráfego legítimo do tráfego potencialmente malicioso. Se uma amostra for classificada como benigna nessa fase, o processo é encerrado. Caso contrário, ela é submetida ao próximo nível de classificação.

O nível 2 tem a função de separar os ataques do tipo *DDoS* dos demais ataques. Durante a operação do modelo, se uma amostra for classificada como *DDoS* nessa fase, o processo é encerrado. Caso contrário, ela é submetida ao terceiro nível de classificação. Os ataques *DDoS* tendem a apresentar padrões de tráfego caracterizados por volumes elevados de pacotes em curtos intervalos de tempo. Esse comportamento volumétrico contrasta fortemente com a maioria dos outros tipos de ataques, que geralmente exploram vulnerabilidades específicas e, portanto, apresentam padrões mais sutis. Dessa forma, esse segundo nível tem o objetivo de aproveitar a característica volumétrica dos ataques *DDoS* para separá-los dos demais ataques, facilitando o processo de classificação e reduzindo a

Algorithm 1: Processo de Treinamento do Modelo BDM

Input: Conjunto de treinamento D
Output: Modelos treinados: ModeloBenigno, ModeloDDoS e ModeloMultiAtaque

```
1
   /* Treinamento do Modelo Benigno                                  */
2  Mapear rótulos de D em Benign e Malicious
3  Treinar ModeloBenigno com D
4
   /* Treinamento do Modelo DDoS                                     */
5  D_mal ← {x ∈ D | y(x) ≠ Benign}
6  Mapear rótulos de D_mal em DDoS e OtherAttacks
7  Treinar ModeloDDoS com D_mal
8
   /* Treinamento do Modelo Multi-Ataque                             */
9  D_outros ← {x ∈ D_mal | y(x) ≠ DDoS}
10 Manter rótulos multiclasse originais de D_outros
11 Treinar ModeloMultiAtaque com D_outros
12
13 return {ModeloBenigno, ModeloDDoS, ModeloMultiAtaque}
```

possibilidade de confusão com outras categorias.

Além disso, ao tratar *DDoS* como uma classe isolada logo após a filtragem de tráfego benigno, é possível mitigar a sobrecarga que esses ataques volumétricos poderiam causar em classificadores posteriores. Isso se torna ainda mais relevante devido ao desbalanceamento do conjunto de dados, no qual os registros de ataques *DDoS* são muito mais numerosos em relação às demais classes. Caso fossem incluídos diretamente em um classificador multiclasse junto aos outros ataques, a predominância de exemplos de *DDoS* poderia enviesar o processo de aprendizado, fazendo com que o modelo atribuísse maior peso a essa classe e, consequentemente, ofuscasse as características discriminativas de ataques menos frequentes. Tal situação comprometeria a capacidade do modelo em detectar adequadamente esses ataques minoritários, o que justificou a adoção dessa etapa intermediária de isolamento dos *DDoS*.

O nível 3 tem como objetivo adicionar a granularidade necessária para identificar os diferentes tipos de ataques que sobraram. Dessa forma, o nível 3 não apenas complementa a filtragem inicial de tráfego malicioso, mas também enriquece a análise, fornecendo maior detalhamento sobre a natureza do ataque detectado, aspecto essencial para subsidiar respostas mais rápidas e eficazes em um ambiente de cibersegurança. As amostras que chegam até esse nível são classificadas como um dos ataques restantes e o processo se encerra. Ao restringir o espaço de decisão apenas a amostras maliciosas não pertencentes à classe *DDoS*, o classificador multiclasse opera em um cenário menos complexo e enviesado, no qual os padrões característicos de cada ataque podem ser explorados de forma mais

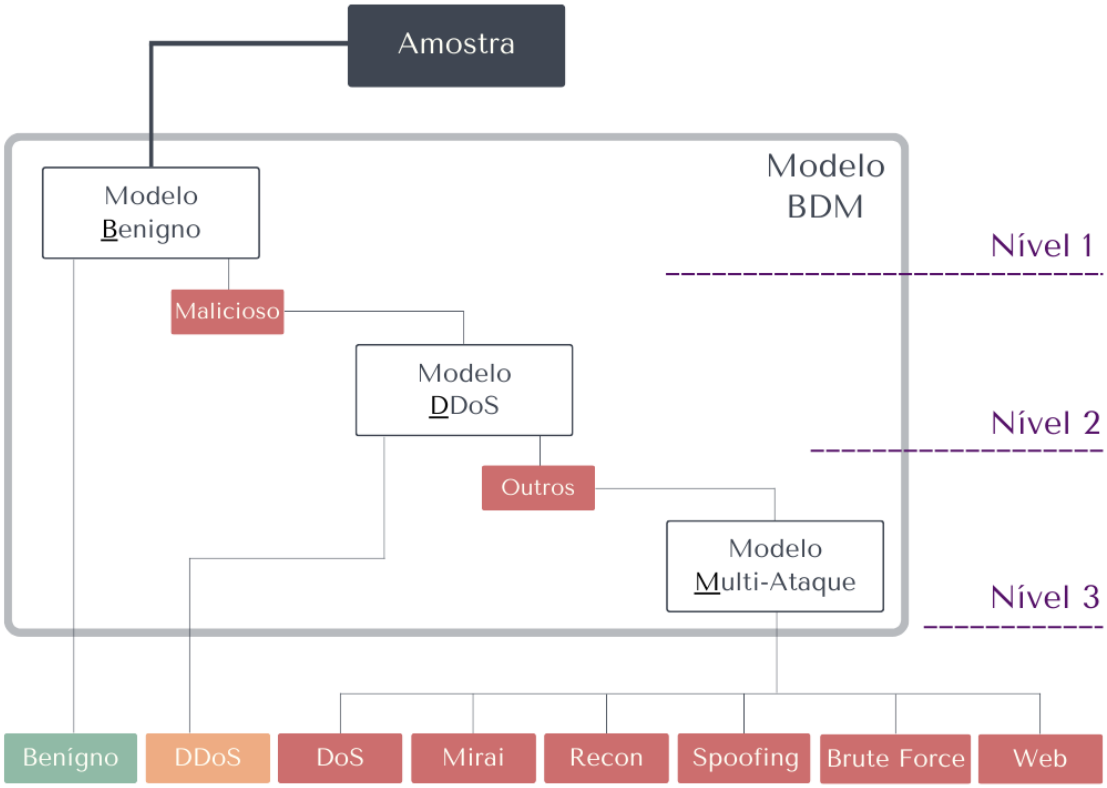

Figura 7 – Fluxo de execução do classificador BDM.

eficaz, sem serem penalizados pelo comportamento dominante dos ataques volumétricos.

Para ilustrar o funcionamento do modelo BDM, o pseudocódigo do processo de classificação é apresentado no Algoritmo 2.

Algorithm 2: Funcionamento do Modelo BDM

Input: Amostra de tráfego x
Output: Classe predita

```
1 if ModeloBenigno(x) = Benign then
2 |   return Benign
3
4 else if ModeloDDoS(x) = DDoS then
5 |   return DDoS
6
7 return ModeloMultiAtaque(x)
```

A entrada do algoritmo corresponde a uma amostra de tráfego de rede IoT, e a saída é a classe predita pelo modelo BDM para essa amostra. As linhas 1 e 2 implementam o nível 1, responsável pela filtragem de tráfego benigno. As linhas 4 e 5 correspondem ao nível 2, que realiza a identificação de ataques do tipo *DDoS*. Por fim, a linha 7 representa

o nível 3, no qual o classificador multiclasse é acionado para distinguir entre os demais tipos de ataques.

Portanto, o modelo hierárquico BDM se mostra uma solução robusta para lidar com os desafios da classificação de tráfego em ambientes IoT. Ao estruturar o processo em três partes distintas - filtragem de tráfego benigno, isolamento de ataques volumétricos e detalhamento multiclasse dos ataques remanescentes - o modelo equilibra simplicidade e granularidade, mitigando problemas recorrentes em abordagens de classificação plana, como a perda de desempenho causada pelo desbalanceamento de classes. Essa hierarquização não apenas melhora a precisão na detecção de tráfego malicioso, mas também garante maior especialização em cada etapa do processo, o que aumenta a interpretabilidade do modelo. Além disso, ao manter a granularidade no terceiro nível, o modelo fornece informações valiosas sobre o tipo específico de ameaça, contribuindo diretamente para estratégias de defesa mais rápidas e direcionadas em um contexto real de cibersegurança. Dessa forma, o modelo BDM representa uma alternativa promissora frente às estratégias convencionais, combinando os benefícios de ambos os modelos binário e multiclasse.

3.4.4 Modelo Hierárquico BRDM

O modelo hierárquico BRDM (**B**enigno, **R**econ, **D**DoS e **M**ulti-Ataque) representa uma extensão direta do modelo BDM, adicionando um nível extra na hierarquia de classificação. Assim como no modelo anterior, o objetivo é combinar a simplicidade do classificador binário com a granularidade de classificadores multiclasse, estruturando o processo de decisão em etapas sucessivas. A diferença central entre os dois está na introdução de um novo classificador específico para os ataques *Recon*, posicionado entre a filtragem de tráfego benigno e o isolamento dos ataques volumétricos *DDoS*. A Figura 8 ilustra o processo de treinamento dos quatro modelos que compõem essa hierarquia.

O modelo BRDM foi projetado com quatro níveis distintos de classificação, cada um composto por um modelo isolado. O nome BRDM surge justamente da junção das primeiras letras dos modelos que o compõem: Benigno, *Recon*, *DDoS* e Multi-Ataque. O processo de treinamento desses modelos segue a mesma ideia do classificador hierárquico anterior, ou seja, para cada modelo são removidas as classes tratadas pelos seus antecessores na hierarquia. Para o treinamento do Modelo *Recon*, todas as amostras benignas foram removidas e todos os ataques diferentes de *Recon* foram agrupados, reorganizando as classes alvos em duas categorias: *Recon* e Outros Ataques. Por sua vez, esse tipo de ataque também será removido dos conjuntos de treino usados para os modelos *DDos* e Multi-Ataque. Esse processo é descrito no Algoritmo 3.

O funcionamento do classificador BRDM, ilustrado na Figura 9, consiste, portanto, na execução sequencial desses quatro modelos.

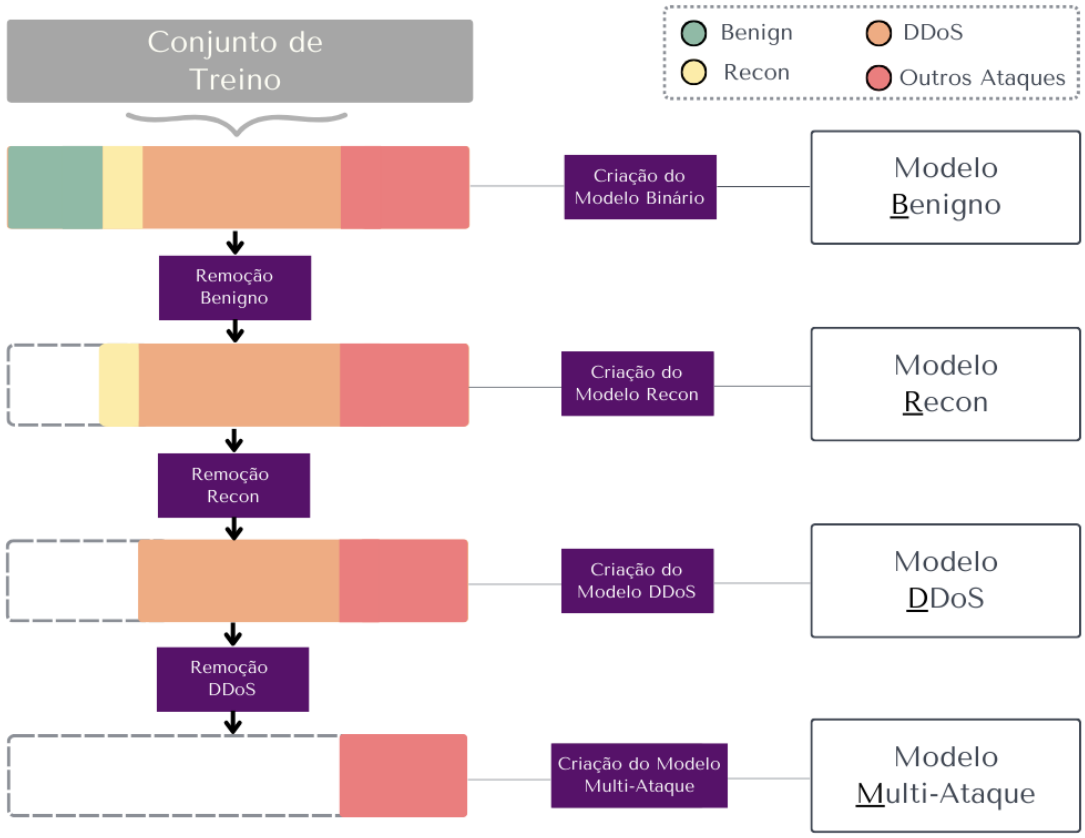

Figura 8 – Fluxo de treinamento dos modelos que compõem o classificador BRDM.

O nível 1 (Modelo Benigno) é idêntico ao do modelo BDM: trata-se de um classificador binário responsável por separar tráfego benigno de tráfego malicioso. Amostras classificadas como benignas encerram o processo de decisão, enquanto as demais seguem para os níveis seguintes.

A principal diferença em relação ao modelo BDM ocorre no nível 2, com a introdução do Modelo *Recon*. Esse classificador binário visa distinguir ataques do tipo *Recon* dos demais ataques. Caso uma amostra seja classificada como *Recon*, o processo de decisão é encerrado. Caso contrário, a amostra segue para o próximo nível.

A motivação por trás da introdução desse nível está no fato de que ataques *Recon* se diferenciam substancialmente dos demais, pois representam uma fase preparatória no ciclo de ataque. Nessa etapa, o agente mal-intencionado coleta informações do alvo - como portas abertas, serviços disponíveis e sistemas em execução - sem executar ações diretamente destrutivas. Essa natureza exploratória torna o tráfego *Recon* distinto, de forma que sua detecção isolada contribui para reduzir a confusão com outras classes de ataque e simplifica os classificadores posteriores. Além disso, sua separação precoce melhora a especialização das fases seguintes, que passam a operar em um espaço de decisão menos heterogêneo.

Algorithm 3: Processo de Treinamento do Modelo BRDM

Input: Conjunto de treinamento D
Output: Modelos treinados: ModeloBenigno, ModeloRecon, ModeloDDoS e ModeloMultiAtaque

1
`/* Treinamento do Modelo Benigno */`
2 Mapear rótulos de D em *Benign* e *Malicious*
3 Treinar ModeloBenigno com D
4
`/* Treinamento do Modelo Recon */`
5 $D_{mal} \leftarrow \{x \in D \mid y(x) \neq Benign\}$
6 Mapear rótulos de D_{mal} em *Recon* e *OtherAttacks*
7 Treinar ModeloRecon com D_{mal}
8
`/* Treinamento do Modelo DDoS */`
9 $D_{outrosRecon} \leftarrow \{x \in D_{mal} \mid y(x) \neq Recon\}$
10 Mapear rótulos de $D_{outrosRecon}$ em *DDoS* e *OtherAttacks*
11 Treinar ModeloDDoS com $D_{outrosRecon}$
12
`/* Treinamento do Modelo Multi-Ataque */`
13 $D_{outrosDDoS} \leftarrow \{x \in D_{outrosRecon} \mid y(x) \neq DDoS\}$
14 Manter rótulos multiclasse originais de $D_{outrosDDoS}$
15 Treinar ModeloMultiAtaque com $D_{outrosDDoS}$
16
17 **return** {ModeloBenigno, ModeloRecon, ModeloDDoS, ModeloMultiAtaque}

O nível 3 corresponde ao Modelo *DDoS*: um classificador binário treinado para distinguir ataques do tipo *DDoS* dos demais ataques. Assim como no modelo hierárquico anterior, se uma amostra for classificada como *DDoS*, o processo é encerrado. Caso contrário, a amostra segue para o nível 4.

Por fim, o nível 4 é composto pelo Modelo Multi-Ataque, responsável por identificar especificamente os ataques restantes (*DoS*, *Mirai*, *Spoofing*, *Web-Based* e *Brute Force*). Ao restringir o espaço de decisão a esse conjunto reduzido, o modelo multiclasse opera em um cenário mais simplificado, explorando com maior precisão as características de cada ataque.

Assim como no BDM, a lógica de operação é simples: uma amostra percorre a hierarquia até ser classificada, com o processo podendo ser encerrado em qualquer um dos níveis. O pseudocódigo desse processo é apresentado no Algoritmo 4.

No algoritmo, a entrada corresponde a uma amostra de tráfego de rede IoT, e a saída é a classe predita pelo modelo BRDM para essa amostra. As linhas 1 e 2 implementam o nível 1, responsável pela filtragem de tráfego benigno. As linhas 4 e 5 correspondem ao nível 2, que realiza a identificação de ataques do tipo *Recon*. As linhas 7 e 8 correspon-

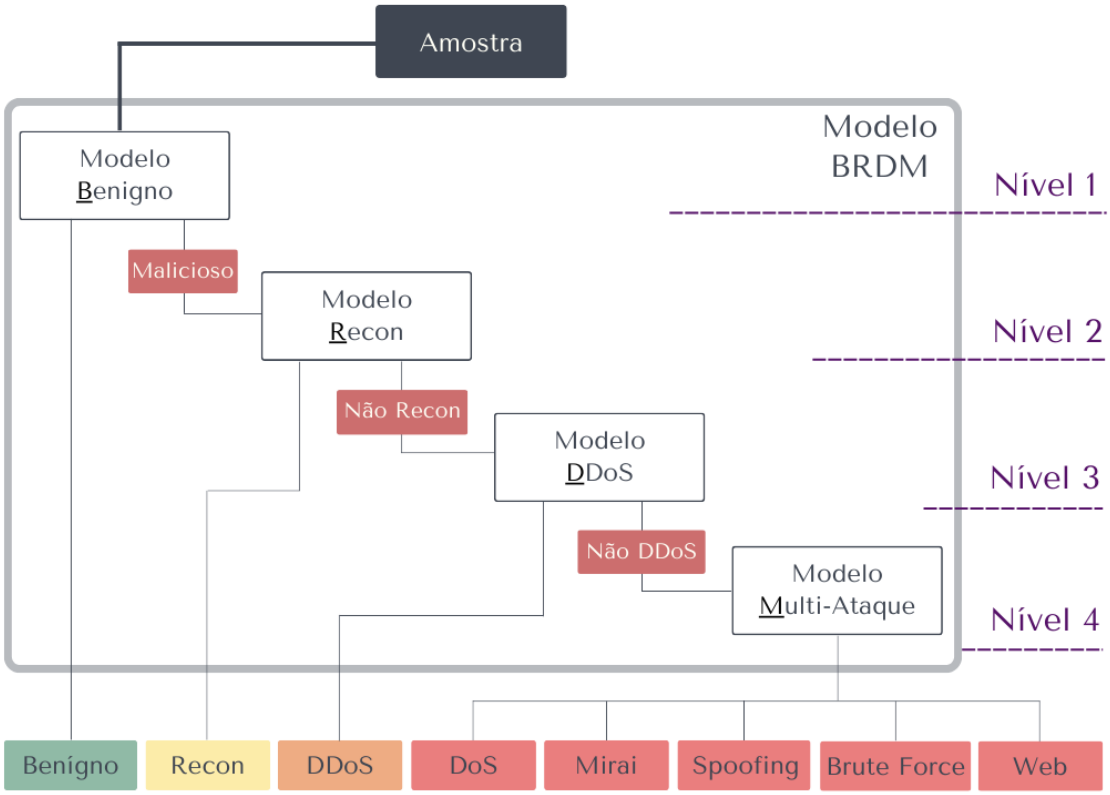

Figura 9 – Fluxo de execução do classificador BRDM.

Algorithm 4: Funcionamento do Modelo BRDM

Input: Amostra de tráfego x
Output: Classe predita

```
1  if ModeloBenigno(x) = Benign then
2  |   return Benign
3
4  else if ModeloRecon(x) = Recon then
5  |   return Recon
6
7  else if ModeloDDoS(x) = DDoS then
8  |   return DDoS
9
10 return ModeloMultiAtaque(x)
```

dem ao nível 3, que realiza a identificação de ataques do tipo *DDoS*. Por fim, a linha 10 representa o nível 4, no qual o classificador multiclasse é acionado para distinguir entre os demais tipos de ataques.

Em síntese, o modelo BRDM difere do BDM pela adição de uma etapa intermediária de isolamento dos ataques *Recon*, aproveitando suas particularidades para melhorar a organização hierárquica do processo de decisão. Essa modificação tem o potencial de

reduzir erros de classificação, simplificar os classificadores subsequentes e oferecer maior desempenho na detecção dos tipos de ataque.

3.5 Avaliação dos Modelos

Após o treinamento, os modelos passam por um processo sistemático de avaliação para validar se o classificador é robusto e confiável o suficiente para ser aplicado em cenários reais. Essa etapa consiste na submissão do classificador a um conjunto de dados formado por amostras que não foram utilizadas no treinamento, para mensurar seu desempenho na tarefa de detecção de intrusões. Por meio dele, é possível medir métricas como acurácia, precisão, *recall* e *F1-score* que permitem, não apenas verificar a capacidade preditiva dos classificadores, mas também identificar eventuais limitações que podem ser melhoradas.

A utilização de amostras não vistas durante o processo de treinamento é fundamental para a avaliação confiável de modelos de aprendizado de máquina. Quando um classificador é testado com dados utilizados no treino, o resultado tende a refletir apenas a sua capacidade de memorizar padrões já conhecidos, sem fornecer indícios claros de sua habilidade de generalização. Por outro lado, avaliar com exemplos inéditos permite verificar se o modelo consegue classificar corretamente novos registros, simulando de maneira mais realista o ambiente de produção em que será aplicado. Essa prática evita problemas como o *overfitting*, em que o modelo apresenta desempenho elevado em dados de treino, mas falha ao lidar com situações não previamente observadas. Portanto, a avaliação é realizada utilizando o conjunto de teste, formado por amostras previamente separadas que não foram utilizadas no treinamento do modelo.

Nos experimentos com classificadores binários e multiclasse, o conjunto de teste é utilizado de forma integral. Entretanto, no caso dos classificadores hierárquicos, foi necessário aplicar um procedimento específico em cada nível da hierarquia. O primeiro nível recebe como entrada todas as amostras do conjunto de teste. Já os níveis subsequentes são avaliados apenas com os registros que foram corretamente classificados no nível anterior e que ainda requerem uma classificação mais refinada. Essa abordagem foi adotada para evitar que os erros de classificação dos níveis anteriores sejam acumulados, prejudicando a avaliação das etapas seguintes da hierarquia. O objetivo é avaliar cada modelo individualmente, considerando somente os erros e acertos do nível em questão. Cabe destacar que a adoção dessa abordagem não compromete a avaliação do erro global dos modelos hierárquicos.

No processo de teste do classificador BDM, inicialmente, o Modelo Benigno é avaliado com o conjunto de teste completo. Em seguida, o Modelo *DDoS* é testado somente com as amostras que foram classificadas como maliciosas e que de fato pertenciam à classe

maliciosa. Dessa forma, exemplos benignos ou falsamente identificados como maliciosos não são considerados nessa etapa. Por fim, o modelo Multi-Ataque é avaliado exclusivamente com as amostras que foram classificadas como 'outros ataques' no nível anterior e que realmente correspondiam a ataques diferentes de *DDoS*. O Algoritmo 5 apresenta, em formato de pseudocódigo, as etapas que compõem o processo de teste do modelo BDM.

Algorithm 5: Processo de Teste do Modelo BDM

Input: Conjunto de teste D
Output: Métricas de desempenho dos três modelos

1
`/* Nível 1: Modelo Benigno */`
2 predBenigno ← classificar(ModeloBenigno, D)
3 resultadoBenigno ← avaliar(predBenigno, D)
4 $D_{mal} \leftarrow \{x \in D \mid predBenigno(x) = Malicious \ \wedge \ y(x) = Malicious\}$
5
`/* Nível 2: Modelo DDoS */`
6 predDDoS ← classificar(ModeloDDoS, D_{mal})
7 resultadoDDoS ← avaliar(predDDoS, D_{mal})
8 $D_{outros} \leftarrow \{x \in D_{mal} \mid predDDoS(x) = OtherAttacks \ \wedge \ y(x) \neq DDoS\}$
9
`/* Nível 3: Modelo Multi-Ataque */`
10 predMulti ← classificar(ModeloMultiAtaque, D_{outros})
11 resultadoMulti ← avaliar(predMulti, D_{outros})
12
13 **return** $\{resultadoBenigno, resultadoDDoS, resultadoMulti\}$

A entrada do Algoritmo 5 corresponde ao conjunto de teste D, e a saída são as métricas de desempenho obtidas em cada nível do modelo BDM. As linhas 2 a 4 implementam o nível 1, no qual o Modelo Benigno é avaliado com todas as amostras do conjunto de teste e, em seguida, é formado o subconjunto D_{mal} contendo apenas as instâncias classificadas como maliciosas que realmente pertencem à classe maliciosa. As linhas 6 a 8 correspondem ao nível 2, no qual o Modelo *DDoS* é avaliado com D_{mal}, sendo então construído o subconjunto D_{outros}, composto apenas por amostras identificadas como "outros ataques" que de fato não pertencem à classe *DDoS*. Por fim, as linhas 10 e 11 implementam o nível 3, no qual o Modelo Multi-Ataque é avaliado exclusivamente com D_{outros}, distinguindo entre os demais tipos de ataques.

De forma análoga, o classificador BRDM segue o mesmo princípio hierárquico, mas com a inclusão do Modelo *Recon* entre o Modelo Benigno e o Modelo *DDoS*. Assim, após a avaliação inicial, o Modelo *Recon* é testado apenas com as amostras corretamente identificadas como maliciosas. Em seguida, o Modelo *DDoS* é avaliado unicamente com as instâncias classificadas como "outros ataques" no nível anterior e que, de fato, correspondiam a ataques diferentes de *Recon*. O Algoritmo 6 apresenta as etapas que compõem o processo de teste do modelo BRDM.

Algorithm 6: Processo de Teste do Modelo BRDM

Input: Conjunto de teste D
Output: Métricas de desempenho dos quatro modelos

1
`/* Nível 1: Modelo Benigno */`
2 predBenigno $\leftarrow$ classificar(ModeloBenigno, D)
3 resultadoBenigno $\leftarrow$ avaliar(predBenigno, D)
4 $D_{mal} \leftarrow \{x \in D \mid predBenigno(x) = Malicious \ \wedge \ y(x) = Malicious\}$
5
`/* Nível 2: Modelo Recon */`
6 predRecon $\leftarrow$ classificar(ModeloRecon, D_{mal})
7 resultadoRecon $\leftarrow$ avaliar(predRecon, D_{mal})
8 $D_{outrosRecon} \leftarrow \{x \in D_{mal} \mid predRecon(x) = OtherAttacks \ \wedge \ y(x) \neq Recon\}$
9
`/* Nível 3: Modelo DDoS */`
10 predDDoS $\leftarrow$ classificar(ModeloDDoS, $D_{outrosRecon}$)
11 resultadoDDoS $\leftarrow$ avaliar(predDDoS, $D_{outrosRecon}$)
12 $D_{outrosDDoS} \leftarrow \{x \in D_{outrosRecon} \mid predDDoS(x) = OtherAttacks \ \wedge \ y(x) \neq DDoS\}$
13
`/* Nível 4: Modelo Multi-Ataque */`
14 predMulti $\leftarrow$ classificar(ModeloMultiAtaque, $D_{outrosDDoS}$)
15 resultadoMulti $\leftarrow$ avaliar(predMulti, $D_{outrosDDoS}$)
16
17 **return** $\{resultadoBenigno, resultadoRecon, resultadoDDoS, resultadoMulti\}$

A entrada do Algoritmo 6 corresponde ao conjunto de teste D, e a saída são as métricas de desempenho obtidas em cada nível do modelo BRDM. As linhas 2 a 4 implementam o nível 1, no qual o Modelo Benigno é avaliado com todas as amostras do conjunto de teste e, em seguida, é formado o subconjunto D_{mal}, contendo apenas as instâncias classificadas como maliciosas que realmente pertencem à classe maliciosa. As linhas 6 a 8 correspondem ao nível 2, no qual o Modelo *Recon* é avaliado com D_{mal}, sendo então gerado o subconjunto $D_{outrosRecon}$, formado apenas pelas amostras identificadas como "outros ataques" que de fato não pertencem à classe *Recon*. As linhas 10 a 12 representam o nível 3, no qual o Modelo *DDoS* é avaliado com $D_{outrosRecon}$, produzindo o subconjunto $D_{outrosDDoS}$, composto apenas por amostras classificadas como "outros ataques" que realmente não pertencem à classe *DDoS*. Por fim, as linhas 14 e 15 implementam o nível 4, no qual o Modelo Multi-Ataque é avaliado exclusivamente com $D_{outrosDDoS}$, distinguindo entre os demais tipos de ataques.

Após a submissão dos modelos ao conjunto de teste, foi construída uma matriz de confusão para cada um deles individualmente, a partir dos resultados de predição, utilizando a função *confusion_matrix* da biblioteca *scikit-learn*. Com base nessas matrizes,

foram calculadas as métricas de avaliação - acurácia, precisão, *recall* e *F1-Score* - por meio das funções *accuracy_score*, *recall_score*, *precision_score* e *f1_score*, disponibilizadas pela mesma biblioteca (Scikit-learn Developers, 2025). Para cada modelo, as métricas foram inicialmente calculadas separadamente para cada classe e, em seguida, consolidadas por meio da média aritmética simples, resultando em um valor global para cada indicador. A opção pelo uso da média simples (*macro average*) justifica-se pelo fato de ela atribuir o mesmo peso a todas as classes, independentemente do número de amostras em cada uma delas, evitando que classes majoritárias dominem a avaliação e permitindo uma análise mais equilibrada e representativa do desempenho dos classificadores em cenários desbalanceados.

Além das métricas de classificação, cada modelo também foi avaliado em relação ao seu desempenho computacional, considerando três aspectos: o tempo total de treinamento, o tempo médio de predição por amostra (em segundos) e o espaço ocupado na memória RAM (em bytes). Esta última medida foi obtida por meio da função *asizeof*, pertencente à biblioteca *pympler* (Pympler Developers, 2024).

Para avaliar o desempenho dos classificadores hierárquicos integralmente, as matrizes de confusão de cada modelo constitutivo foram examinadas a fim de avaliar a eficácia na filtragem progressiva dos ataques, identificar confusões entre diferentes categorias e contabilizar os erros e acertos do classificador como um todo em relação a cada classe-alvo. A partir dessa consolidação, foi calculado o *recall* de cada classe, visando mensurar a capacidade do classificador hierárquico em identificar corretamente cada tipo de tráfego. A Eq. 3.1 mostra como é feito o cálculo do *recall* para cada uma das classes:

$$\text{Recall} = \frac{TotalAcertos}{TotalAmostras} \tag{3.1}$$

4 Experimentos

Neste capítulo, são apresentados e discutidos os resultados obtidos a partir da execução dos experimentos conduzidos com os modelos propostos. Inicialmente, descreve-se a configuração do ambiente experimental, contemplando os recursos computacionais e as decisões tomadas nesse processo. Em seguida, são analisados os desempenhos dos classificadores, tanto de forma individual quanto no contexto das arquiteturas hierárquicas, buscando evidenciar pontos fortes, limitações e padrões de comportamento observados.

4.1 Ambiente Experimental

Esta seção descreve os procedimentos e recursos empregados para a realização dos experimentos, com o objetivo de garantir a reprodutibilidade dos resultados e fornecer transparência quanto às condições em que os classificadores foram avaliados. Para tanto, são detalhados os critérios de divisão do conjunto de dados, a escolha e configuração do algoritmo de classificação, as bibliotecas utilizadas e o ambiente computacional em que as execuções foram conduzidas.

No processo de pré-processamento e transformação dos dados, foram utilizadas principalmente as bibliotecas *pandas* e *scikit-learn*. A biblioteca *pandas* foi empregada no tratamento inicial do conjunto, destacando-se o uso das funções *dropna()* e *drop_duplicates()* para a remoção de registros ausentes e duplicados, assegurando maior integridade e consistência (Pandas Developers, 2025). A biblioteca *scikit-learn*, por sua vez, foi utilizada na padronização dos atributos contínuos, por meio da função *StandardScaler()*, que ajusta os valores para média zero e variância unitária (Scikit-learn Developers, 2025).

A divisão estratificada do conjunto de dados em treino e teste foi realizada por meio da função *train_test_split*, também disponibilizada pela biblioteca *scikit-learn*. Neste trabalho, adotou-se a proporção de 70% para treinamento e 30% para teste. Para mitigar possíveis vieses decorrentes dessa divisão, o processo foi repetido dez vezes, e os experimentos foram avaliados individualmente em cada repetição. Dessa forma, o resultado final reportado foi calculado por meio da média aritmética simples entre os resultados dos dez experimentos.

Todos os classificadores foram implementados utilizando o algoritmo *Random Forest*, por meio da implementação disponível na biblioteca *scikit-learn* (versão 1.6.1). A escolha desse algoritmo fundamenta-se na revisão bibliográfica feita por esta pesquisa, que o destaca como técnica eficaz para tarefas de classificação no contexto de detecção de intrusões em redes IoT. Para assegurar a comparabilidade entre os modelos, os hiperpa-

râmetros do algoritmo foram mantidos em suas configurações padrão.

Na etapa de avaliação, a biblioteca *scikit-learn* foi utilizada para a construção das matrizes de confusão, por meio da função *confusion_matrix*, e para o cálculo das métricas de desempenho - acurácia, precisão, *recall* e *F1-Score* - com as funções *accuracy_score*, *precision_score*, *recall_score* e *f1_score*. As métricas foram inicialmente calculadas para cada classe e, em seguida, consolidadas por meio da média aritmética simples (*macro average*), garantindo que todas as classes recebessem o mesmo peso, independentemente de sua representatividade no conjunto de dados. Além disso, a biblioteca *pympler* foi empregada para mensurar o espaço ocupado pelos modelos na memória RAM, por meio da função *asizeof*, complementando a análise de desempenho computacional.

Os experimentos foram conduzidos no ambiente de desenvolvimento *Jupyter Notebook*, utilizando a linguagem *Python* (versão 3.12.7) (THOMAS et al., 2016). Esse ambiente foi escolhido pela praticidade na prototipação, pela integração com bibliotecas de aprendizado de máquina e pelas facilidades na visualização e interpretação dos resultados. As execuções foram realizadas em uma máquina equipada com processador *Intel(R) Core(TM) Ultra 7 155H* (3.80 GHz) e 32 GB de memória RAM instalada. O sistema operacional adotado foi o *Windows 11 Home*.

4.2 Avaliação dos Classificadores Planos

Como ponto de partida, foram conduzidos experimentos com os classificadores planos: binário e multiclasse. O objetivo dessa etapa é estabelecer uma base de comparação, de modo a analisar o desempenho de ambos em condições idênticas às dos experimentos com os classificadores hierárquicos. A partir dessa análise preliminar, busca-se compreender as vantagens e limitações inerentes a cada abordagem, bem como verificar se os classificadores hierárquicos conseguem incorporar os benefícios de ambos, mitigando suas respectivas desvantagens.

A Tabela 4 apresenta as métricas de desempenho preditivo calculadas para ambos os classificadores, considerando a média de dez execuções independentes dos experimentos. Esses resultados representam a consolidação do desempenho dos classificadores planos e servem como referência para a análise comparativa com os modelos hierárquicos.

Tabela 4 – Resultado das métricas de desempenho preditivo para os classificadores binário e multiclasse.

	Binário	**Multiclasse**
Acurácia	0,9962	0,9941
Precisão	0,9694	0,9641
Recall	0,9814	0,8338
F1	0,9753	0,8777

Do ponto de vista das métricas clássicas de avaliação, o classificador binário apresentou desempenho superior ao multiclasse em praticamente todos os indicadores. Os dois modelos tiveram uma acurácia semelhante, com vantagem discreta para o binário que alcançou 99,62%, contra 99,41% do multiclasse. Entretanto, o classificador binário destacou-se pelo equilíbrio entre precisão (96,94%) e *recall* (98,14%), resultando em um F1-Score de 97,53%. Isso evidencia que o modelo não apenas foi confiável ao indicar uma instância como positiva (alta precisão), mas também conseguiu identificar a grande maioria das instâncias que realmente pertenciam à classe positiva (alto *recall*). Já o classificador multiclasse obteve precisão relativamente elevada (96,41%), mas apresentou *recall* consideravelmente inferior (83,39%), resultando em um *F1-Score* de 87,77%. Em termos práticos, isso revela que o modelo multiclasse, apesar de evitar muitos falsos positivos, deixou de detectar uma parcela considerável das instâncias que deveriam ser classificadas corretamente, comprometendo sua eficácia em um cenário de detecção de intrusões, no qual falsos negativos são especialmente críticos por representarem ataques não identificados.

Para compreender de forma mais detalhada os padrões de erro e acerto dos classificadores planos, as Figuras 10 e 11 apresentam suas respectivas matrizes de confusão. Essas matrizes correspondem a uma das dez execuções realizadas, considerando o mesmo particionamento entre treino e teste, e permitem visualizar como cada modelo se comportou na distinção entre as classes, evidenciando os casos de acerto e as principais confusões observadas.

A análise das matrizes de confusão revela diferenças importantes no comportamento dos classificadores. O classificador binário apresentou melhor desempenho na detecção de ataques, identificando corretamente 4.847.884 das 4.859.998 instâncias maliciosas, deixando de detectar 12.114 (falsos negativos). Já o multiclasse obteve desempenho inferior nesse aspecto, detectando 4.843.575 ataques e deixando de identificar 16.423, o que representa um aumento na taxa de falsos negativos. Por outro lado, na detecção de tráfego benigno, o multiclasse apresentou desempenho superior, classificando corretamente 193.761 das 197.672 instâncias, contra 190.902 do binário, resultando em menor número de falsos positivos (3.911 contra 6.770). Esses resultados indicam que, enquanto o classificador binário é mais eficaz na detecção de ataques, o multiclasse demonstra ser melhor na detecção de tráfego benigno, reduzindo alarmes falsos. Em cenários reais de sistemas de segurança, no entanto, o custo de não detectar um ataque (falso negativo) tende a ser mais crítico do que o de gerar um alarme falso (falso positivo), uma vez que uma intrusão não identificada pode comprometer seriamente a integridade e a disponibilidade da rede. Nesse contexto, ainda que o multiclasse ofereça vantagens na redução de falsos alarmes, o classificador binário se mostra mais adequado para aplicações práticas em detecção de intrusões, onde a prioridade é maximizar a sensibilidade do sistema a possíveis ataques.

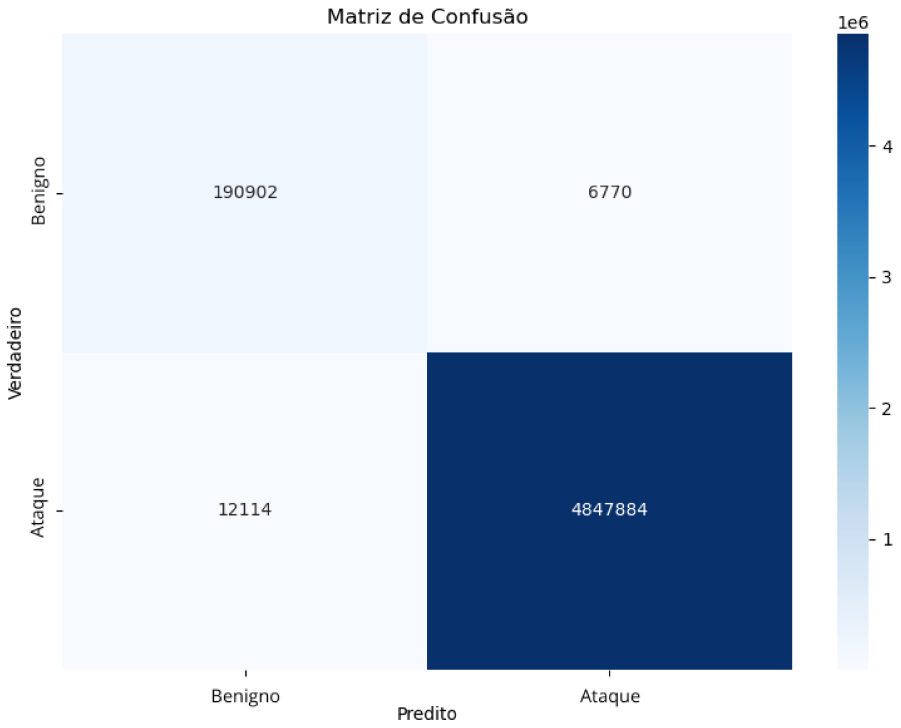

Figura 10 – Matriz de confusão do classificador binário.

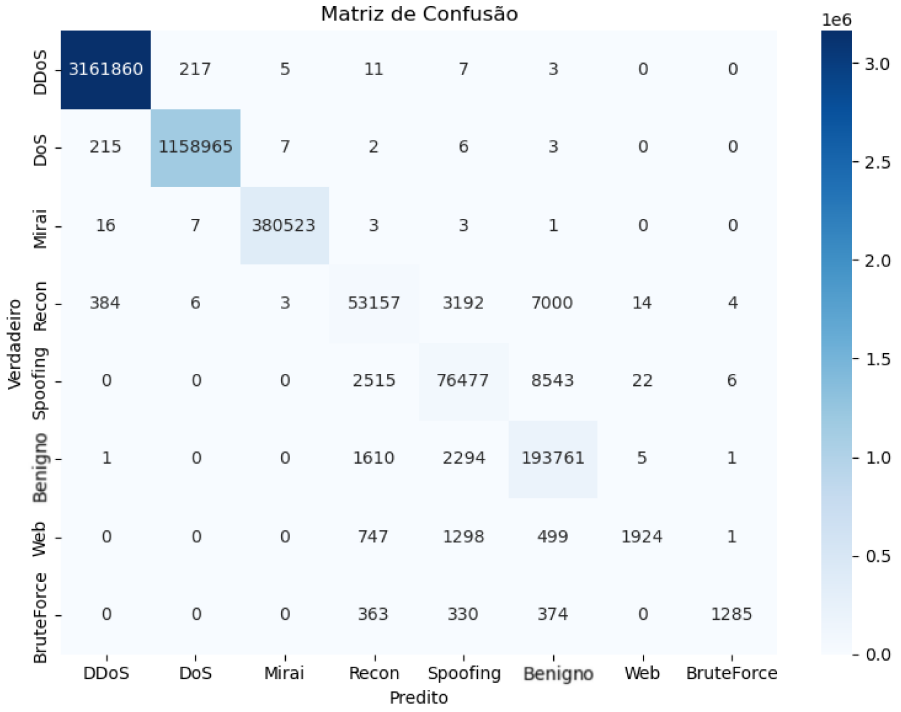

Figura 11 – Matriz de confusão do classificador multiclasse.

Contudo, a filtragem simples entre ataques e tráfego normal não é a única questão relevante para a cibersegurança. Outra habilidade essencial nessa área é a capacidade de distinguir entre diferentes tipos de ataque, característica unicamente presente no classificador multiclasse. Portanto, apesar de apresentar menor sensibilidade na detecção global de intrusões, esse modelo oferece uma vantagem estratégica ao permitir a identificação específica das categorias de ataque, fornecendo informações valiosas para a tomada de decisão e para a definição de medidas de mitigação mais adequadas. Assim, verifica-se um *trade-off* entre os dois classificadores planos: enquanto o binário se mostra mais confiável para garantir a detecção de intrusões, o multiclasse contribui com maior granularidade na caracterização das ameaças, aspecto que pode ser determinante em cenários práticos de resposta a incidentes.

Nesse sentido, também é essencial avaliar o desempenho do classificador multiclasse na identificação de cada tipo de ataque e compreender as principais confusões entre categorias. A matriz de confusão mostra que as classes majoritárias, como *DDoS*, *DoS* e *Mirai*, alcançaram taxas de acerto bastante elevadas: no caso de *Mirai*, 380.523 das 380.553 instâncias foram corretamente classificadas (apenas 30 erros); para *DoS*, 1.158.965 de 1.159.198 (233 erros); e para DDoS, 3.161.860 de 3.162.103 (243 erros). Esses valores indicam que, para classes amplamente representadas no conjunto de dados, os erros são residuais em relação ao volume total. Em contraste, as classes minoritárias apresentaram maior dispersão de erros, refletindo a dificuldade do modelo em diferenciá-las. No caso de *Recon*, por exemplo, mais de 10 mil instâncias foram atribuídas incorretamente a outras classes, sobretudo *Benign* e *Spoofing*. De forma semelhante, em *Spoofing*, milhares de instâncias foram confundidas com *Recon* ou *Benign*. Já os ataques Web e BruteForce mostraram sobreposição considerável com *Benign*, *Spoofing* e *Recon*, reforçando a dificuldade na separação precisa dessas categorias. Esses resultados sugerem que, embora o classificador multiclasse seja capaz de diferenciar entre tipos específicos de ataques, essa habilidade é limitada, especialmente nas classes minoritárias, onde o desbalanceamento dos dados amplifica as confusões.

Com o objetivo de aprofundar a compreensão sobre as confusões entre classes ao longo das dez execuções realizadas, foi elaborada uma tabela que consolida o desempenho do classificador em cada categoria alvo. Nessas tabelas, são apresentados o total de instâncias, a quantidade de acertos e erros, bem como o valor de *recall* correspondente, permitindo uma análise detalhada do comportamento do modelo por classe. Esses valores foram calculados pela média das dez execuções. A Tabela 5 apresenta essa análise para o classificador binário, enquanto a Tabela 6 reúne os resultados obtidos pelo classificador multiclasse.

No classificador binário, a detecção de ataques foi de 4.847.885 para uma média de 4.859.998 ataques, resultando em 12.113 amostras incorretamente classificadas como be-

Tabela 5 – Análise das classes alvo do classificador binário.

	Total de Registros	**Acertos**	**Erros**	**Recall**
Malicioso	4859998	4847885	12113	0,9975
Benigno	197672	190831	6841	0,9653

Tabela 6 – Análise das classes alvo do classificador multiclasse.

	Total de Registros	**Acertos**	**Erros**	**Recall**
Mirai	380553	380525	28	0,9999
DDoS	3162103	3161856	247	0,9999
DoS	1159198	1158953	245	0.9997
Benigno	197672	193657	4015	0,9796
Spoofing	87563	76298	11265	0,8713
Recon	63760	53353	10407	0,8367
BruteForce	2352	1286	1066	0,5467
Web	4469	1954	2515	0,4370

nignas. Esse desempenho corresponde a um *recall* de 99,75% para a classe de ataques. Já o classificador multiclasse obteve desempenho inferior nesse aspecto, com 4.834.225 ataques corretamente identificados, o que praticamente dobrou o número de erros (25.773) e reduziu o *recall* para 99,46%. Por outro lado, o multiclasse apresentou melhor desempenho na detecção de tráfego benigno, atingindo um *recall* de 97,96%, contra 96,53% observado no binário. Ademais, no contexto multiclasse, verificou-se uma dificuldade na classificação de algumas categorias de ataques específicas: as classes *Web* e *Brute Force* apresentaram os menores desempenhos, com valores de *recall* inferiores a 55%, seguidas pelas classes *Recon* (83,67%) e *Spoofing* (87,13%). Em contrapartida, as classes *Mirai*, *DDoS* e *DoS* exibiram elevadas taxas de acerto, com *recalls* superiores a 99%, demonstrando baixa taxa de confusão.

Além das métricas de desempenho preditivo, foi conduzida também uma análise sob a perspectiva computacional, com o intuito de verificar a viabilidade prática dos classificadores planos em cenários reais. Essa análise contemplou aspectos relacionados ao tempo de treinamento, ao tempo médio de teste por amostra e ao espaço ocupado pelos modelos na memória RAM. A Tabela 7 apresenta os valores médios obtidos para cada métrica, considerando dez execuções independentes para ambos os classificadores.

Tabela 7 – Resultado das métricas de desempenho computacional para os classificadores binário e multiclasse.

	Binário	**Multiclasse**
Tempo de Treino	39,21 min	64,70 min
Tempo de Teste	0,005 ms	0,011 ms
Tamanho	123.464 bytes	128.344 bytes

Sob a perspectiva computacional, o classificador binário se mostrou mais eficiente.

Seu tempo médio de treinamento foi de aproximadamente 39 minutos, valor significativamente inferior ao registrado pelo multiclasse de aproximadamente 64 min, indicando menor custo de processamento durante a fase de construção do modelo. Em termos de tempo de predição, ambos apresentaram tempos extremamente reduzidos, mas novamente o binário foi mais eficiente, com um tempo médio de 5 microssegundos por amostra, contra 11 microssegundos do multiclasse. Quanto ao tamanho em memória RAM, os dois modelos mostraram-se semelhantes, com vantagem discreta para o binário (123.464 bytes) em relação ao multiclasse (128.344 bytes). Assim, os resultados computacionais reforçam a superioridade do classificador binário em cenários que exigem rapidez na etapa de treinamento e maior economia de recursos.

Em síntese, a análise dos classificadores planos evidenciou um claro *trade-off* entre desempenho preditivo e granularidade da detecção. O classificador binário destacou-se pela elevada sensibilidade na identificação de ataques, reduzindo a probabilidade de falsos negativos, característica essencial em cenários de cibersegurança, onde a omissão de uma intrusão pode comprometer gravemente a rede. Por outro lado, o classificador multiclasse, ainda que tenha apresentado menor *recall* global, mostrou-se mais eficaz na identificação de tráfego benigno e, sobretudo, oferece a capacidade de distinguir entre diferentes tipos de ataque, fornecendo informações valiosas para estratégias de mitigação direcionadas. Sob a perspectiva computacional, o classificador binário também se mostrou mais eficiente, exigindo menor tempo de treinamento e apresentando uma leve vantagem no uso de recursos, embora ambos os modelos tenham se mostrado viáveis em termos de execução. Diante disso, torna-se evidente a necessidade de uma abordagem que combine a alta sensibilidade do classificador binário com a capacidade de categorização do multiclasse.

4.3 Avaliação do Classificador BDM

Diante das limitações observadas nos experimentos com os classificadores planos, este trabalho propôs um classificador hierárquico denominado BDM, estruturado em três modelos distintos: Benigno, DDoS e Multi-Ataque. Essa abordagem busca combinar as vantagens da classificação binária e multiclasse, de modo a maximizar a detecção de intrusões e, ao mesmo tempo, fornecer uma caracterização mais detalhada das diferentes categorias de ataques.

A avaliação do classificador BDM foi conduzida em três etapas complementares. Na primeira, analisou-se o desempenho de cada modelo individualmente, utilizando métricas tradicionais de classificação - acurácia, precisão, *recall* e *F1-Score* -, com o objetivo de verificar sua eficácia isolada. Em seguida, investigou-se o funcionamento do classificador como um todo, por meio da análise das matrizes de confusão correspondentes a cada nível hierárquico, permitindo compreender a qualidade da filtragem realizada, bem como os

erros e confusões entre classes após o processo completo de decisão. Além disso, essa etapa também investigou mais a fundo os erros, acertos e revocação de cada uma das classes após passar por todos os níveis do BDM. Por fim, realizou-se uma análise computacional dos modelos isoladamente, considerando o tempo de treinamento, o tempo médio de teste por amostra e o espaço ocupado na memória RAM, fatores relevantes para avaliar sua viabilidade prática em cenários reais.

4.3.1 Desempenho Individual dos Modelos

Primeiramente, para cada um dos modelos - Benigno, *DDoS* e Multi-Ataque - foram calculadas as métricas tradicionais de avaliação individualmente. Vale lembrar que, para fins de análise, os erros de modelos de níveis superiores na hierarquia não foram propagados para o restante dos modelos, visando que essas métricas refletissem a qualidade de cada modelo isoladamente. Dessa forma, as métricas apresentadas não devem ser interpretadas de maneira comparativa entre os modelos, mas sim como uma caracterização do desempenho específico de cada etapa da hierarquia. A Tabela 8 apresenta os resultados médios obtidos por cada um dos modelos, considerando dez execuções independentes dos experimentos.

Tabela 8 – Resultado das métricas de desempenho preditivo para os modelos que compõem o classificador BDM.

	Benigno	**DDoS**	**Multi-Ataque**
Acurácia	0,9962	0,9998	0,9938
Precisão	0,9694	0,9998	0,9678
Recall	0,9814	0,9998	0,8176
F1	0,9753	0,9998	0,8671

Como o processo de treinamento do classificador binário tradicional é realizado da mesma forma que no modelo Benigno, uma vez que ambos possuem o mesmo objetivo (separação entre tráfego benigno e malicioso), seus resultados foram diretamente aproveitados para a execução do primeiro nível da hierarquia. Dessa forma, as métricas obtidas são idênticas, com acurácia de 99,62%, precisão de 96,94%, *recall* de 98,14% e *F1-Score* de 97,53%. Esses valores evidenciam desempenho consistente, com bom equilíbrio entre a correta identificação de instâncias legítimas e a baixa taxa de falsos alarmes, aspectos essenciais em sistemas de detecção em tempo real.

No caso do modelo *DDoS*, o desempenho manteve-se elevado, com todas as métricas acima de 99,98%. Esses resultados refletem a facilidade do modelo em reconhecer ataques dessa categoria, possivelmente em razão de duas características principais: a natureza volumétrica desse tipo de ataque, cujo padrão de tráfego difere de forma significativa de outros ataques, e a forte representatividade da classe no conjunto de dados, o que favorece o aprendizado do modelo.

Por fim, o modelo Multi-Ataque, responsável pela classificação multiclasse entre *DoS*, *Mirai*, *Recon*, *Spoofing*, *Web-Based* e *Brute Force*, apresentou acurácia de 99,38% e precisão de 96,78%. Entretanto, o valor de *recall* foi consideravelmente mais baixo (81,76%), resultando em um *F1-Score* de 86,71%. Esses valores revelam desequilíbrio entre as métricas, indicando que o modelo, apesar de assertivo ao identificar uma instância como pertencente a uma classe de ataque (alta precisão), falhou em reconhecer uma parcela relevante das ocorrências reais, o que compromete sua sensibilidade. Esse comportamento reflete não apenas a maior complexidade inerente à tarefa de diferenciar múltiplos tipos de ataques, mas também o desbalanceamento das classes, que limita a capacidade do modelo em aprender padrões de categorias minoritárias (como *Web-Based* e *Brute Force*).

4.3.2 Análise do Classificador Integrado

A avaliação do classificador BDM como sistema integrado foi conduzida a partir da análise das matrizes de confusão de cada nível da hierarquia, de modo a compreender o processo completo de filtragem adotado por esse classificador. As matrizes apresentadas correspondem à mesma execução utilizada na análise dos classificadores planos, assegurando a consistência do particionamento entre treino e teste.

Para possibilitar uma avaliação mais aprofundada, os experimentos incluíram uma estrutura complementar que manteve, para cada amostra, a associação com sua classe original de tráfego. Esse procedimento permitiu contabilizar, em todos os níveis, os acertos e erros por categoria, mesmo nos casos em que a classificação era binária e não distinguia explicitamente entre os diferentes tipos de ataque.

Assim, a análise não se restringe às matrizes de confusão, mas incorpora também tabelas que consolidam o desempenho por classe original, contemplando o total de instâncias, número de acertos, número de erros e valor de *recall* obtido. Para assegurar a robustez dos resultados, tais valores foram calculados individualmente em cada execução experimental e, ao final, consolidados por meio da média aritmética das dez execuções realizadas.

O ponto de partida desta análise é o modelo benigno, correspondente ao primeiro nível da hierarquia, responsável pela separação entre tráfego benigno e malicioso. A Figura 12 apresenta a matriz de confusão correspondente, enquanto a Tabela 9 detalha os resultados obtidos para cada classe original. Essa análise evidencia a qualidade da filtragem inicial realizada pelo modelo, permitindo verificar sua capacidade em distinguir tráfego legítimo de instâncias de ataque e fornecendo a base para as etapas subsequentes.

Como o modelo benigno foi construído a partir do mesmo processo que criou o classificador binário, suas matrizes de confusão são idênticas. Esse resultado evidencia

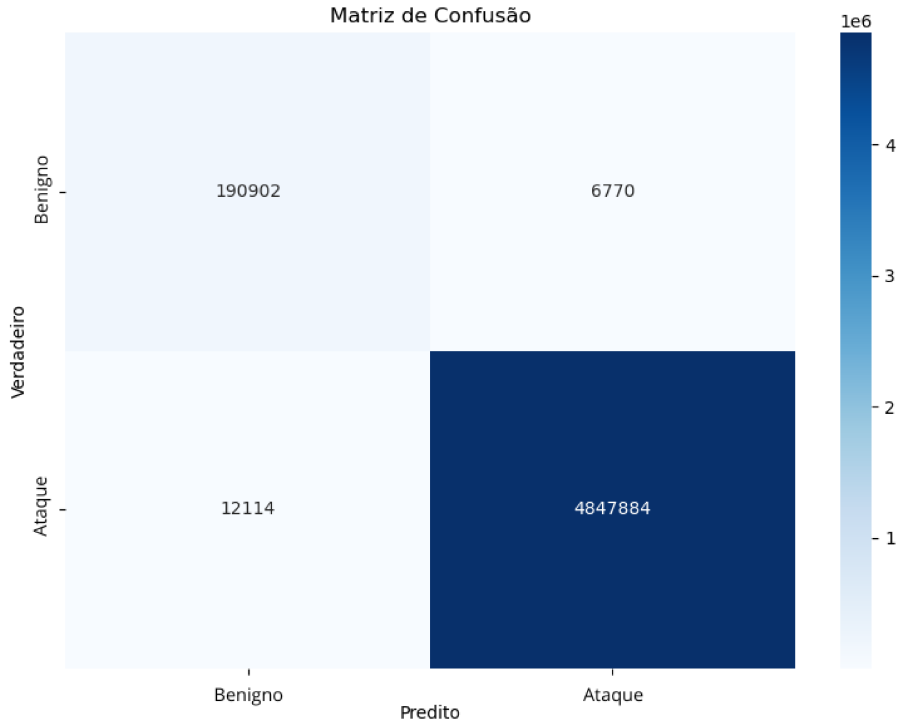

Figura 12 – Matriz de confusão do modelo Benigno BDM.

que, ao iniciar a hierarquia pela filtragem de tráfego benigno, o classificador BDM herda a elevada sensibilidade do binário na identificação de ataques, reduzindo a probabilidade de falsos negativos e atendendo a um dos objetivos principais deste experimento. Nesse nível, o BDM identificou corretamente 4.847.884 das 4.859.998 instâncias maliciosas, apresentando 12.114 falsos negativos. Para o tráfego benigno, classificou corretamente 190.902 das 197.672 instâncias, desempenho equivalente ao do classificador binário, mas inferior ao multiclasse, que obteve 193.761 acertos. Considerando que, em sistemas de detecção de intrusões, o custo de não detectar um ataque tende a ser mais crítico do que a geração de alarmes falsos, esse *trade-off* favorece a abordagem adotada pelo classificador BDM.

A Tabela 9 complementa essa análise ao indicar quais classes de ataque apresentaram maior confusão com o tráfego benigno ao longo das dez execuções, permitindo identificar categorias mais suscetíveis a escapar da detecção e, portanto, representar maiores riscos aos usuários. Nesse contexto, para cada classe de ataque, o número de acertos corresponde às instâncias corretamente classificadas como maliciosas, enquanto os erros indicam aquelas equivocadamente atribuídas como benignas. Os resultados apresentados correspondem à média das dez execuções realizadas, de modo a assegurar maior robustez nas análises.

Assim como no classificador binário, considerando uma média de 4.859.998 ataques avaliados por execução, 4.847.885 foram corretamente classificados como tráfego malicioso, resultando em um *recall* de 99,75% ao considerar apenas a filtragem de ataques. Entre os

Tabela 9 – Análise das classes originais para o nível 1 do classificador BDM.

	Total de Registros	**Acertos**	**Erros**	**Recall**
DoS	1159198	1159197	1	0,9999
Mirai	380553	380552	1	0,9999
DDoS	3162103	3162100	3	0,9999
Benigno	197672	190831	6841	0,9653
Web	4469	4312	157	0,9648
Spoofing	87563	80962	6601	0.9246
Recon	63760	58603	5157	0,9191
BruteForce	2352	2159	193	0,9179

0,25% ataques não detectados (12.113 registros incorretos), destacaram-se as categorias *Brute Force*, *Recon* e *Spoofing* como as mais propensas a serem confundidas com tráfego benigno, seguidas pela classe *Web-Based*. Em contrapartida, os ataques *DoS*, *Mirai* e *DDoS* apresentaram valores de *recall* superiores a 99,99%, demonstrando baixíssima taxa de confusão com tráfego legítimo já nesse primeiro nível da hierarquia. Além disso, 96,53% das instâncias benignas foram corretamente identificadas, evidenciando um desempenho consistente na detecção de tráfego legítimo, ainda que inferior ao obtido pelo classificador multiclasse.

As instâncias corretamente classificadas como maliciosas no primeiro nível são encaminhadas ao segundo, cujo objetivo é identificar ataques do tipo *DDoS*. É importante destacar que, para simplificação da avaliação, as amostras incorretamente classificadas como maliciosas na etapa anterior não prosseguem na hierarquia, sendo contabilizadas apenas no desempenho do modelo Benigno e do classificador BDM como um todo. A Figura 13 apresenta a matriz de confusão referente ao modelo *DDoS*, enquanto a Tabela 10 detalha o desempenho em relação a cada classe original. Essa etapa permite avaliar a capacidade do modelo em reconhecer ataques volumétricos em meio às demais categorias de intrusão.

A matriz de confusão revela a facilidade do modelo DDoS BDM em separar ataques do tipo DDoS, identificando corretamente 3.161.810 das 3.162.102 instâncias dessa classe que passaram para esse nível, deixando de detectar apenas 292 amostras dessa categoria. O desempenho foi praticamente equivalente ao do classificador multiclasse, que acertou 3.161.860 instâncias, superando o modelo *DDoS* por apenas 50 amostras, diferença desprezível frente ao volume total avaliado. Quanto à confusão de outros tipos de ataques com *DDoS*, o modelo hierárquico apresentou desempenho ligeiramente superior, registrando 471 erros contra 615 do multiclasse.

A Tabela 10 complementa a análise ao indicar quais categorias de ataque apresentaram maior confusão com *DDoS* ao longo das dez execuções. Para os ataques diferentes de *DDoS*, os acertos correspondem ao número de instâncias corretamente atribuídas à ca-

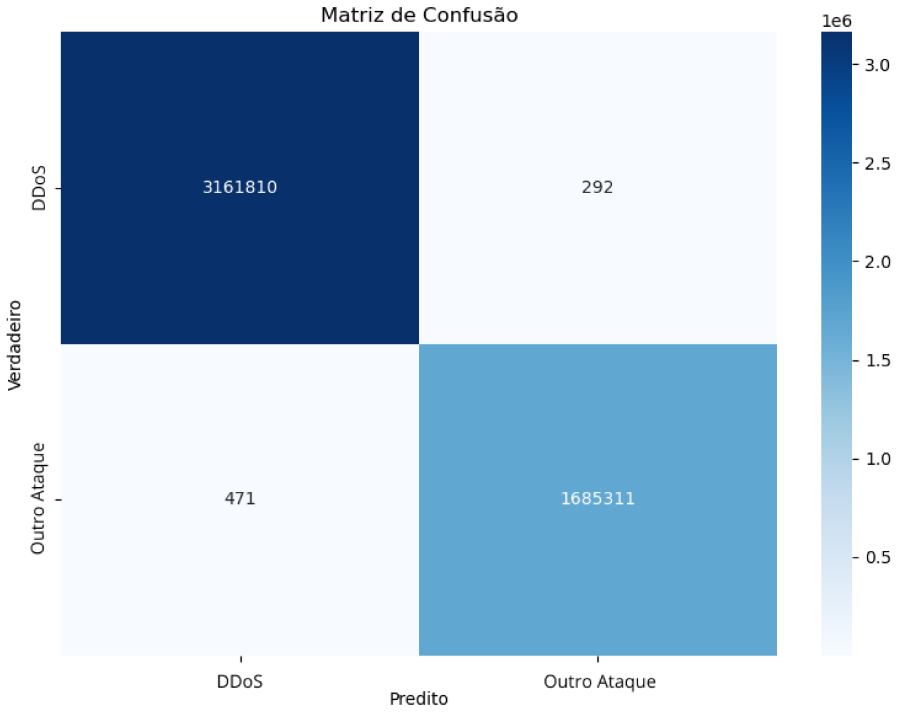

Figura 13 – Matriz de confusão do modelo DDoS BDM.

tegoria "Não *DDoS*", enquanto os erros indicam o número de instâncias equivocadamente classificadas como *DDoS*.

Tabela 10 – Análise das classes originais para o nível 2 do classificador BDM.

	Total de Registros	**Acertos**	**Erros**	**Recall**
BruteForce	2159	2159	0	1,0000
Web	4312	4312	0	1,0000
Spoofing	80962	80962	0	1,0000
Mirai	380552	380550	2	0,9999
DDoS	3162100	3161804	296	0,9999
DoS	1159197	1158993	204	0,9998
Recon	58603	58375	228	0,9961

Considerando a média das execuções, de um total de 3.162.100 ataques *DDoS*, 3.161.804 foram corretamente reconhecidos, resultando em um *recall* de 99,99% ao considerar apenas a filtragem desse tipo de ataque. Quanto às demais classes, apenas 434 das 1.685.785 instâncias foram incorretamente atribuídas a *DDoS*, evidenciando baixo nível de confusão. Ao longo das dez execuções, ataques *Brute Force*, *Web-Based* e *Spoofing* não foram confundidos em nenhuma ocasião, alcançando *recall* perfeito. Já as classes *DoS*, *Mirai* e *DDoS* também apresentaram níveis baixos de confusão, todas com *recall* superior a 99,96%. A única exceção foi a categoria *Recon*, que obteve o menor índice, com *recall* de 99,61%, ainda assim considerado um desempenho bastante satisfatório.

Por fim, as amostras corretamente classificadas como não DDoS no segundo nível são encaminhadas ao terceiro nível da hierarquia, destinado à diferenciação entre as demais categorias de ataque (*DoS*, *Mirai*, *Recon*, *Spoofing*, *Web-Based* e *Brute Force*). Assim como no cenário anterior, as instâncias incorretamente classificadas como não DDoS não prosseguem, sendo computadas nos erros do segundo nível. A Figura 14 apresenta a matriz de confusão obtida nesse estágio, e a Tabela 11 consolida os resultados por classe original. Essa etapa é particularmente relevante por envolver a tarefa mais complexa do processo hierárquico, permitindo identificar tanto as categorias em que há alta confiabilidade quanto aquelas mais suscetíveis a confusões.

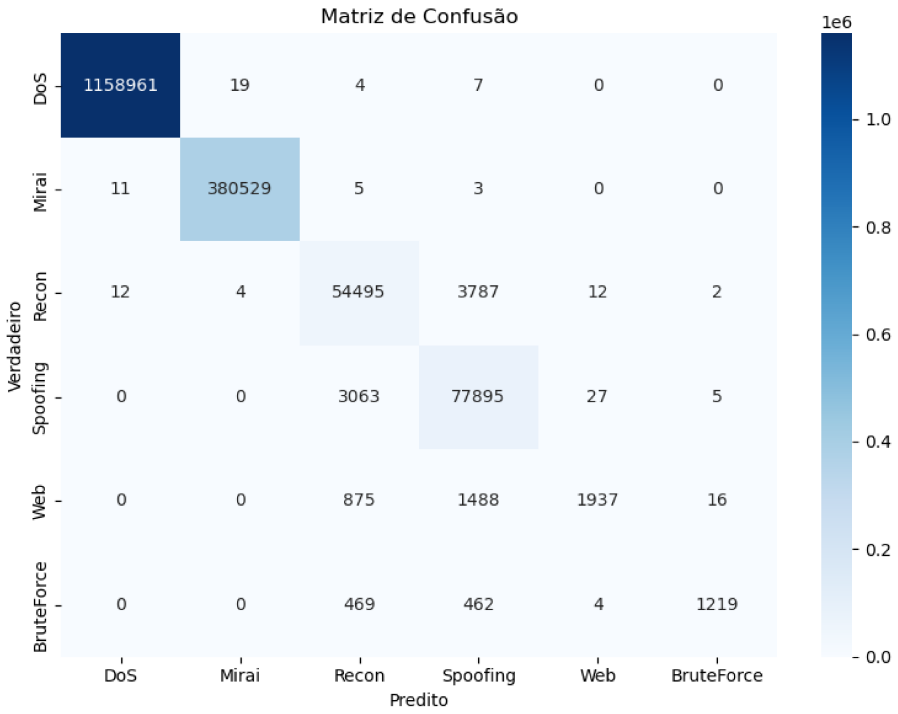

Figura 14 – Matriz de confusão do modelo Multi-Ataque BDM.

Analisando a matriz de confusão, o ataque *DoS* apresentou 1.158.961 instâncias corretamente classificadas, com apenas 30 erros distribuídos entre outras categorias. De forma semelhante, a classe *Mirai* alcançou 380.529 acertos, com apenas 19 erros, confirmando também uma taxa de detecção excelente.

No caso da classe *Recon*, observa-se um maior nível de confusão, com 54.495 instâncias corretamente identificadas, mas 3.817 classificadas incorretamente, sobretudo como *Spoofing*. Já a classe *Spoofing* demonstrou resultados robustos, com 77.895 acertos e apenas 3.095 erros, a maioria confundidos com *Recon*, o que aponta uma dificuldade de diferenciar essas duas categorias. Apesar disso, o modelo ainda atingiu um desempenho consistente para essas classes.

As classes minoritárias, por sua vez, apresentaram maior suscetibilidade a erros

de classificação. A categoria *Web-Based* teve 1.937 instâncias corretamente classificadas, mas 2.379 incorretamente atribuídas a outras classes, principalmente *Recon* e *Spoofing*. De maneira semelhante, a classe *Brute Force* alcançou apenas 1.219 acertos, com 935 erros distribuídos entre diferentes categorias, especialmente *Recon* e *Spoofing*, evidenciando maior fragilidade na sua detecção.

Em síntese, a análise da matriz de confusão revela que o modelo multi-ataque do BDM apresenta excelente desempenho nas classes majoritárias, como *DoS* e *Mirai*, além de resultados sólidos em *Spoofing* e *Recon*. Contudo, as categorias menos representadas no conjunto de dados, como *Web-Based* e *Brute Force*, demonstraram maior dificuldade de detecção, indicando um impacto direto do desbalanceamento das classes no desempenho do classificador. Outro aspecto relevante é a elevada taxa de confusão de diversas categorias com os ataques *Spoofing* e *Recon*, o que evidencia uma sobreposição de padrões entre essas classes e aponta para um possível ponto de aprimoramento a ser explorado.

Ao comparar o modelo multi-ataque do BDM com o classificador multiclasse plano, observa-se que ambos apresentam elevado desempenho nas classes majoritárias, como *DoS* e *Mirai*, com taxas de acerto praticamente equivalentes. Nas classes minoritárias, como *Web-Based* e *Brute Force*, os dois demonstraram dificuldades semelhantes, com baixas taxas de acerto e elevada dispersão dos erros entre diferentes categorias. O ataque *Brute Force* foi a única categoria que obteve menos acertos no modelo multi-ataque (1.219 acertos) do que no classificador multiclasse (1.285 acertos). Em contrapartida, o BDM apresentou desempenho superior nas classes *Recon* (54.495 acertos contra 53.157) e *Spoofing* (77.895 acertos contra 76.477), evidenciando ganhos pontuais nessas categorias. De forma geral, o classificador BDM alcançou um resultado semelhante ao do classificador multiclasse na distinção de tipos de ataques nessa execução, cumprindo, assim, um dos objetivos centrais desta pesquisa.

Para sustentar essa análise, a Tabela 11 contabiliza a média de acertos e erros por categoria para o modelo multi-ataque ao longo de dez execuções.

Tabela 11 – Análise das classes originais para o nível 3 do classificador BDM.

	Total de Registros	**Acertos**	**Erros**	**Recall**
Mirai	380550	380531	19	0,9999
DoS	1158993	1158965	28	0,9999
Spoofing	80962	77763	3199	0,9604
Recon	58375	54601	3774	0,9353
BruteForce	2159	1207	952	0,5590
Web	4312	1945	2367	0,4510

A análise da tabela evidencia que o desempenho do modelo apresenta forte dependência da representatividade de cada classe no conjunto de dados. As classes majoritárias, como *DoS* e *Mirai*, alcançaram *recall* de 99,99%, com pouquíssimos erros (28 e 19, respec-

tivamente), confirmando a elevada capacidade de detecção para categorias amplamente representadas. As classes intermediárias, *Recon* e *Spoofing*, também demonstraram resultados satisfatórios, com *recall* de 93,53% e 96,04%, respectivamente, embora ainda apresentem confusões relevantes - especialmente no caso da classe *Recon*, que registrou 3.774 erros. Em contrapartida, observa-se um desempenho significativamente inferior nas classes minoritárias: *Brute Force* atingiu apenas 55,90% de *recall*, enquanto *Web-Based* foi a mais prejudicada, com apenas 45,10% de *recall*.

Contabilizando o total de erros e acertos em cada um dos níveis, durante as dez execuções, foi possível montar uma tabela para o classificador BDM como um todo, analisando o seu desempenho por classe. Esses resultados são mostrados na Tabela 12.

Tabela 12 – Análise das classes alvo do classificador BDM.

	Total de Registros	**Acertos**	**Erros**	**Recall**
Mirai	380553	380531	22	0,9999
DDoS	3162103	3161805	298	0,9999
DoS	1159198	1158965	233	0,9997
Benigno	197672	190831	6841	0,9653
Spoofing	87563	77763	9800	0,8880
Recon	63760	54601	9159	0,8563
BruteForce	2352	1208	1144	0,5136
Web	4469	1945	2524	0,4352

A Tabela referente ao classificador BDM como um todo evidencia um desempenho bastante elevado para as classes majoritárias. As categorias *DDoS*, *DoS* e *Mirai* apresentaram *recall* próximo de 100%, confirmando a capacidade do classificador em lidar com ataques volumétricos e altamente representados no conjunto de dados. O tráfego benigno também foi identificado com alta taxa de acerto, atingindo 96,53% de *recall*, desempenho equivalente ao observado no classificador binário. Em contrapartida, as classes minoritárias continuaram sendo as mais desafiadoras: *Brute Force* obteve apenas 51,36% de *recall*, enquanto *Web-Based* apresentou 43,52%, revelando elevada taxa de confusão e confirmando o impacto do desbalanceamento de classes na hierarquia. Já as classes intermediárias, *Recon* e *Spoofing*, alcançaram *recalls* de 85,63% e 88,80%, respectivamente, indicando um desempenho satisfatório, mas apontando dificuldades na distinção entre categorias semanticamente próximas. Esses resultados evidenciam que, embora o BDM tenha mantido desempenho elevado para classes majoritárias e tráfego benigno, sua sensibilidade permanece limitada em categorias pouco representadas.

Ao comparar o desempenho do classificador BDM com o modelo multiclasse plano, observa-se que ambos apresentaram resultados praticamente equivalentes nas classes majoritárias, como *DDoS*, *DoS* e *Mirai*, todas com *recall* próximo de 100%. Para o tráfego *Benign*, o modelo multiclasse alcançou desempenho superior (*recall* de 97,96% contra 96,53% do BDM), reduzindo significativamente a quantidade de falsos positivos. Já nas classes mi-

noritárias, os resultados foram similares, com ligeira vantagem para o multiclasse para os ataques *Brute Force* (*recall* de 54,67% contra 51,36% do BDM) e *Web-Based* (43,52% contra 43,70%, diferença desprezível). Em relação às categorias intermediárias, como *Recon* e *Spoofing*, o BDM obteve melhor desempenho, com *recalls* de 85,63% e 88,80%, respectivamente, contra 83,67% e 87,13% no modelo multiclasse. Esses resultados sugerem que a hierarquia do BDM não trouxe ganhos expressivos nas classes majoritárias nem nas minoritárias, mas demonstrou maior capacidade de distinção em categorias intermediárias, reforçando sua viabilidade como alternativa estrutural para classificação de ataques em ambientes IoT. Dessa forma, conclui-se que o classificador BDM conseguiu alcançar a capacidade do classificador multiclasse de distinguir entre múltiplos tipos de ataques, apresentando desempenho muito próximo e até mesmo superior em algumas categorias.

Portanto, além de alcançar um *recall* de 99,75% na filtragem de ataques, equivalente ao desempenho obtido pelo classificador binário, o classificador BDM também demonstrou capacidade de distinguir entre diferentes tipos de intrusão de forma semelhante ao classificador multiclasse. Dessa forma, o BDM cumpre o objetivo central desta pesquisa ao combinar, em uma única abordagem hierárquica, as principais qualidades dos dois classificadores planos: alta sensibilidade na detecção de ataques e capacidade de discriminar múltiplas categorias de ameaças.

4.3.3 Avaliação Computacional

Por fim, para completar a análise do classificador BDM, foi feita uma avaliação em termos de desempenho computacional. A Tabela 13 apresenta os resultados obtidos por cada um dos modelos, considerando uma média de dez execuções independentes.

Tabela 13 – Resultado das métricas de desempenho computacional para os modelos que compõem o classificador BDM.

	Benigno	**DDoS**	**Multi-Ataque**
Tempo de Treino	39,21 min	53,83 min	16,21 min
Tempo de Teste	0,005 ms	0,007 ms	0,007 ms
Tamanho	123.464 bytes	123.464 bytes	126.696 bytes

No que se refere ao desempenho computacional dos modelos isolados, observa-se que os modelos Benigno e *DDoS* apresentaram tempos de treinamento mais elevados, de aproximadamente 39,21 e 53,83 minutos, respectivamente, além de ocuparem o mesmo tamanho em memória RAM (123.464 bytes). Já o modelo Multi-Ataque demandou significativamente menos tempo de treinamento, cerca de 16,21 minutos, embora apresente tamanho superior (126.696 bytes). Em termos de tempo de teste por amostra, todos os modelos mostraram desempenho bom, variando entre 0,005 e 0,007 milissegundos.

O tempo de treinamento mais elevado observado nos modelos *DDoS* e Benigno,

em comparação ao modelo Multi-Ataque, pode ser explicado pelo forte desbalanceamento do conjunto de dados, no qual a classe *DDoS* é amplamente majoritária. Esse cenário faz com que esses dois modelos precisem processar um volume muito maior de instâncias durante a etapa de aprendizado, aumentando a quantidade de operações necessárias para o ajuste dos parâmetros internos. Assim, o tempo de treinamento não reflete apenas a arquitetura do modelo, mas também a distribuição das instâncias no conjunto de dados, sendo diretamente impactado pela predominância da classe *DDoS*.

Para fins de simplificação, o desempenho computacional do classificador BDM como sistema integrado foi estimado a partir da soma dos custos de seus três níveis hierárquicos, desconsiderando o eventual *overhead* associado ao tráfego de dados entre os modelos. Embora tal custo exista, presume-se que seu impacto não seja significativo no desempenho global. O tempo médio de treinamento foi de aproximadamente 109 minutos, valor consideravelmente superior ao observado nos classificadores planos - 39 minutos no binário e 64 minutos no multiclasse. Esse aumento, entretanto, é restrito à fase de construção do modelo e, portanto, não compromete sua utilização prática em produção. Durante a inferência, o tempo médio de teste por amostra foi de 19 microssegundos, superior ao binário (5 microssegundos) e ao multiclasse (11 microssegundos), mas ainda suficientemente baixo para aplicações em tempo real. Em termos de memória, o BDM apresentou tamanho de aproximadamente 373.624 bytes, significativamente maior que os classificadores planos - 123.464 bytes no binário e 128.344 bytes no multiclasse.

Esses aumentos de tempo e memória são esperados, uma vez que o classificador BDM envolve a execução sequencial de três modelos distintos. Tal complexidade adicional, no entanto, é compensada pela capacidade do classificador em combinar os pontos fortes das abordagens binária e multiclasse, atendendo ao objetivo central desta investigação. Ainda assim, os resultados apontam oportunidades para pesquisas futuras voltadas à redução dos custos computacionais do BDM, de modo a torná-lo mais eficiente sem comprometer sua robustez na classificação hierárquica.

4.4 Avaliação do Classificador BRDM

Diante dos erros de predição observados na avaliação do classificador BDM, foi proposto um segundo classificador hierárquico, denominado BRDM, que incorpora um nível adicional voltado à identificação de ataques do tipo *Recon*. A principal motivação para essa nova abordagem foi a elevada taxa de confusão envolvendo os ataques de reconhecimento, o que comprometeu a precisão na distinção entre certas categorias. A avaliação do BRDM seguiu a mesma metodologia adotada para o BDM, sendo estruturada em três etapas: análise individual dos modelos, investigação do comportamento do classificador integrado e avaliação do desempenho computacional.

4.4.1 Desempenho Individual dos Modelos

Para avaliar a qualidade preditiva dos modelos que compõem o classificador BRDM, foram calculadas métricas tradicionais de desempenho de forma individualizada. Assim como no caso do BDM, os erros provenientes de modelos de níveis superiores da hierarquia não foram propagados para os demais, de modo que os resultados refletem exclusivamente a capacidade de cada modelo isoladamente, em seus respectivos cenários de classificação. Assim, os valores reportados não têm como finalidade a comparação direta entre as etapas, mas sim a descrição do desempenho alcançado por cada nível da hierarquia de forma independente. A Tabela 14 apresenta os resultados médios obtidos em dez execuções.

Tabela 14 – Resultado das métricas de desempenho preditivo para os modelos que compõem o classificador BRDM.

	Benigno	**Recon**	**DDoS**	**Multi-Ataque**
Acurácia	0,9962	0,9984	0,9999	0,9986
Precisão	0,9694	0,9719	0,9998	0,9897
Recall	0,9814	0,9604	0,9998	0,8595
F1	0,9753	0,9661	0,9998	0,9089

O modelo benigno do classificador BRDM opera da mesma forma que no classificador BDM. Assim, como esperado, ele obteve os mesmos valores observados no classificador binário e no modelo benigno do BDM, confirmando a consistência e robustez desse modelo para discriminação entre instâncias benignas e maliciosas.

O modelo *Recon*, incluído especificamente no BRDM para lidar com as confusões observadas entre ataques de reconhecimento e outras categorias, apresentou métricas bastante satisfatórias. Com acurácia de 99,84%, precisão de 97,19%, *recall* de 96,05% e *F1-Score* de 96,61%, esse modelo demonstrou boa capacidade de identificar ataques *Recon*, equilibrando baixo índice de falsos positivos e elevada taxa de detecção.

O modelo DDoS manteve desempenho excepcional, com todas as métricas próximas de 100%. A acurácia foi de 99,99%, enquanto precisão, *recall* e *F1-Score* ficaram em 99,98%. Esse desempenho é consistente com os resultados observados no BDM e reforça a facilidade de distinguir ataques volumétricos de DDoS em relação a outras categorias de tráfego, tanto pela clara diferença nos padrões de tráfego quanto pela forte representatividade dessa classe no conjunto de dados.

Por fim, o modelo Multi-Ataque, responsável por classificar simultaneamente *DoS*, *Mirai*, *Spoofing*, *Web-Based* e *Brute Force*, apresentou resultados superiores aos obtidos na última etapa do classificador BDM. Tal resultado era esperado uma vez que a remoção prematura dos ataques do tipo *Recon* reduziu a complexidade desse modelo. O recall foi de 85,95%, evidenciando uma taxa de detecção satisfatória, por mais que ainda apresente dificuldades em reconhecer todas as instâncias reais das classes minoritárias. A precisão

foi significativamente mais alta (98,97%), resultando em um *F1-Score* de 90,89%, o que aponta um baixo índice de falsos positivos e um bom equilíbrio entre a precisão e o *recall*. Esses valores indicam que, apesar da complexidade da tarefa multiclasses e do desbalanceamento entre categorias, o modelo conseguiu uma boa assertividade geral.

4.4.2 Análise do Classificador Integrado

A avaliação do classificador BRDM como sistema integrado seguiu a mesma metodologia adotada para o BDM, considerando as matrizes de confusão de cada nível hierárquico e tabelas que consolidam o desempenho por classe original. As matrizes apresentadas correspondem à mesma execução utilizada na análise dos classificadores analisados anteriormente, com o mesmo particionamento entre treino e teste. Assim como no caso anterior, cada execução experimental manteve a associação de todas as instâncias com suas classes de origem, possibilitando o cálculo dos acertos, erros e valores de *recall* por categoria, mesmo em níveis de classificação binária.

Dessa forma, a análise do BRDM buscou compreender não apenas o desempenho de cada modelo isolado, mas também a contribuição de cada estágio da hierarquia para o processo completo de filtragem. O ponto de partida continua sendo o modelo Benigno, seguido pela etapa de detecção específica de ataques de *Recon*, e posteriormente pelos classificadores *DDoS* e Multi-Ataque. Essa estrutura permite avaliar em que medida a inclusão de um nível intermediário dedicado a ataques de reconhecimento contribui para reduzir confusões observadas na abordagem BDM.

Começando pelo modelo Benigno, a Figura 15 apresenta a matriz de confusão correspondente, enquanto a Tabela 15 detalha os resultados obtidos para cada classe original. É importante destacar que, embora o processo de treinamento seja equivalente ao realizado no classificador binário tradicional (separação entre tráfego benigno e malicioso), os resultados obtidos não foram idênticos aos observados no modelo BDM. Isso ocorreu porque, neste experimento, não foi possível reutilizar diretamente o classificador binário como foi feito para o BDM. Assim, foi necessário treinar um novo modelo e, devido ao caráter estocástico do algoritmo *Random Forest*, pequenas variações nos resultados finais foram inevitáveis.

Analisando a matriz, o BRDM classificou corretamente 4.847.940 das 4.859.998 instâncias maliciosas, resultando em 12.058 falsos negativos. Para o tráfego benigno, foram corretamente identificadas 190.937 das 197.672 instâncias. Dessa forma, o modelo apresentou desempenho equivalente ao classificador binário tradicional na filtragem de ataques, reduzindo a ocorrência de falsos negativos e atendendo a um dos principais objetivos deste trabalho. A Tabela 15 complementa essa análise ao avaliar esse desempenho ao longo das dez execuções.

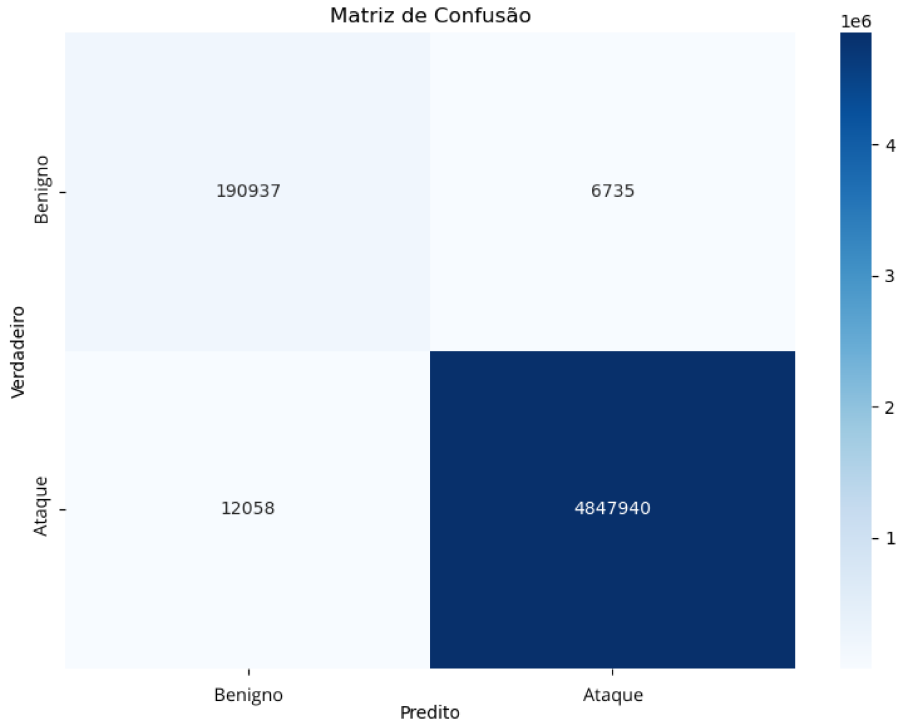

Figura 15 – Matriz de confusão do modelo Benigno BRDM.

Tabela 15 – Análise das classes originais para o nível 1 do classificador BRDM.

	Total de Registros	**Acertos**	**Erros**	**Recall**
DoS	1159198	1159197	1	0,9999
Mirai	380553	380552	1	0,9999
DDoS	3162103	3162100	3	0,9999
Benigno	197672	190828	6844	0,9653
Web	4469	4314	155	0,9653
Spoofing	87563	80955	6608	0,9245
Recon	63760	58608	5152	0,9191
BruteForce	2352	2160	192	0,9183

De acordo com os resultados, de um total médio de 4.859.998 instâncias de ataque, 4.847.886 foram corretamente identificadas como tráfego malicioso, resultando em um *recall* de 99,75% para a filtragem de ataques. Esse valor é equivalente ao obtido tanto pelo classificador binário tradicional quanto pelo modelo BDM. Embora a distribuição dos erros entre as classes apresente pequenas diferenças em relação aos dois classificadores anteriores, tais variações decorrem unicamente da natureza estocástica do algoritmo *Random Forest*. Na prática, o desempenho dos classificadores hierárquicos BDM e BRDM é o mesmo neste nível da hierarquia.

Diferente do classificador BDM, as instâncias corretamente classificadas como maliciosas no primeiro nível são encaminhadas para o modelo *Recon*, cujo objetivo é identificar

e separar ataques de reconhecimento em meio às demais categorias de intrusão. A Figura 16 apresenta a matriz de confusão referente ao modelo, enquanto a Tabela 16 detalha o desempenho em relação a cada classe original.

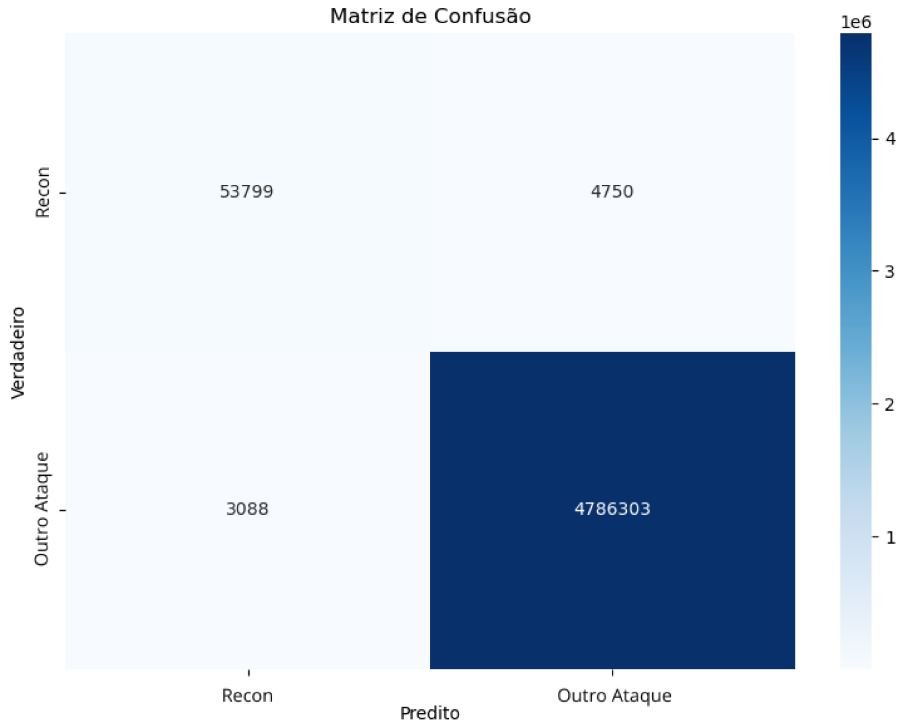

Figura 16 – Matriz de confusão do modelo Recon BRDM.

A matriz de confusão evidencia que o classificador foi capaz de identificar corretamente 53.799 instâncias da classe *Recon*, enquanto 4.750 amostras foram incorretamente rotuladas como pertencentes a outras categorias de ataque. O número de acertos dessa classe superou por pouco o classificador multiclasse (53.157 acertos), mas piorou consideravelmente quando comparado com o classificador BDM (54.495 acertos). Por outro lado, entre as instâncias de ataques não pertencentes à classe *Recon*, 4.786.303 foram corretamente classificadas, com apenas 3.088 sendo equivocadamente atribuídas à classe *Recon*, superando ambos os classificadores multiclasse (3.641 confusões) e BDM (4.416 confusões mesmo desconsiderando erros da classe DDoS). Tais resultados demonstram que o modelo apresenta boa capacidade de discriminação, especialmente na rejeição de instâncias não relacionadas a *Recon*, contribuindo para reduzir ambiguidades observadas no classificador anterior.

A Tabela 16 complementa a análise ao indicar quais categorias de ataque apresentaram maior confusão com *Recon* ao longo das dez execuções. Para os ataques diferentes de *Recon*, acertos correspondem ao número de instâncias corretamente atribuídas à categoria "Não *Recon*", enquanto erros indicam o número de instâncias equivocadamente classificadas como *Recon*.

Tabela 16 – Análise das classes originais para o nível 2 do classificador BRDM.

	Total de Registros	**Acertos**	**Erros**	**Recall**
DoS	1159197	1159196	1	0,9999
Mirai	380552	380551	1	0,9999
DDoS	3162100	3162096	4	0,9999
Spoofing	80955	78621	2334	0,9711
Recon	58608	54016	4592	0,9216
Web	4314	3799	515	0,8806
BruteForce	2160	1857	303	0,8597

Considerando a média das execuções, de um total de 58.608 ataques *Recon*, 54.016 foram corretamente reconhecidos, resultando em um *recall* de 92,16% ao considerar apenas a filtragem desse tipo de ataque. Quanto às demais classes, apenas 3.158 das 4.789.278 instâncias foram incorretamente atribuídas a *Recon* (99,93% de acerto para outros ataques), evidenciando baixo nível de confusão. As maiores taxas de confusão ocorreram nos ataques *Brute Force* e *Web-Based*, seguidos por *Spoofing*, enquanto as categorias *DoS*, *Mirai* e *DDoS* apresentaram valores de *recall* superiores a 99,99%, evidenciando baixíssima suscetibilidade a erros de classificação nesse nível.

As instâncias corretamente classificadas como não *Recon* no segundo nível são encaminhadas ao terceiro, cujo objetivo é identificar ataques do tipo *DDoS*. A Figura 17 apresenta a matriz de confusão referente ao modelo *DDoS*, enquanto a Tabela 17 detalha o desempenho em relação a cada classe original. Essa etapa permite avaliar a capacidade do classificador em reconhecer ataques volumétricos em meio às demais categorias de intrusão, já excluídas as instâncias de *Recon*.

A matriz de confusão demonstra a elevada eficácia do modelo, que identificou corretamente 3.161.831 das 3.162.098 instâncias da classe *DDoS* encaminhadas a este nível, resultando em apenas 267 falsos negativos. Esse desempenho é comparável ao obtido pelo classificador BDM, evidenciando consistência nos resultados. Em relação às demais categorias de ataque, a confusão com *DDoS* mostrou-se mínima: apenas 215 instâncias foram incorretamente atribuídas a essa classe em um universo de 1.624.205 amostras, o que confirma a robustez do modelo na rejeição de falsos positivos. A Tabela 17 complementa essa análise ao indicar em detalhe quais categorias apresentaram maior propensão a serem confundidas com tráfego de ataque distribuído.

Considerando a média das execuções, de um total de 3.162.096 ataques *DDoS*, 3.161.825 foram corretamente reconhecidos, resultando em um *recall* de 99,99% para essa categoria. Quanto às demais classes, apenas 207 das 1.278.025 instâncias foram incorretamente atribuídas a *DDoS* (99,98% de acerto para outros ataques), o que demonstra baixo nível de confusão. Observa-se ainda que os ataques *Brute Force*, *Spoofing* e *Web-Based* não apresentaram nenhum erro de classificação, alcançando *recall* perfeito. As classes *Mi-*

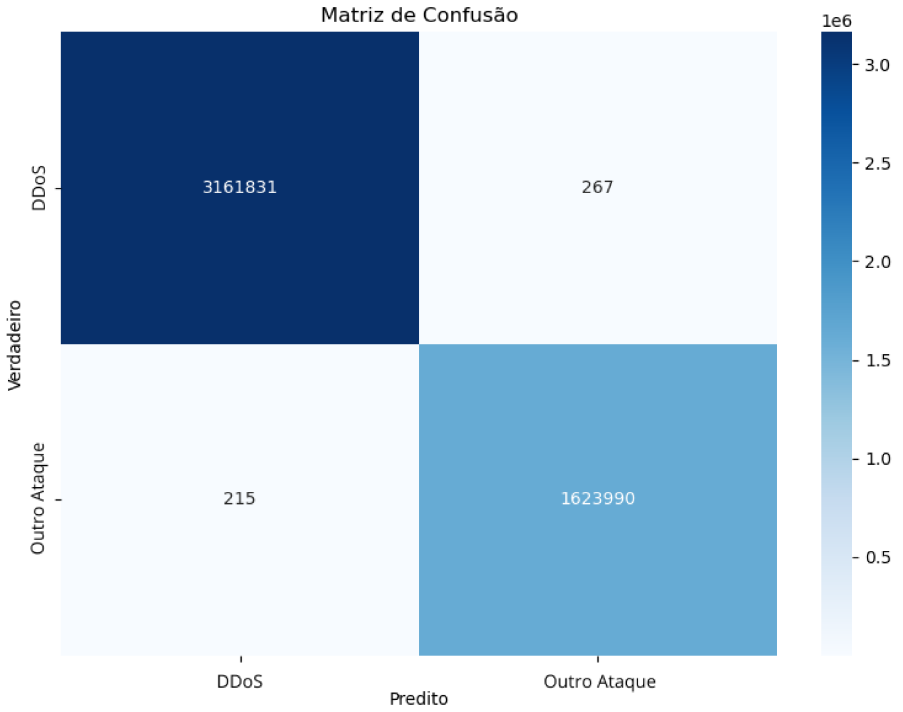

Figura 17 – Matriz de confusão do modelo DDoS BRDM.

Tabela 17 – Análise das classes originais para o nível 3 do classificador BRDM.

	Total de Registros	**Acertos**	**Erros**	**Recall**
BruteForce	1857	1857	0	1,0000
Spoofing	78621	78621	0	1,0000
Web	3799	3799	0	1,0000
Mirai	380552	380548	4	0,9999
DDoS	3162096	3161825	271	0,9999
DoS	1159196	1158993	203	0,9998

rai e *DoS*, por sua vez, registraram pequenas quantidades de confusão, mas mantiveram desempenho elevado, com *recall* de 99,99% e 99,98%, respectivamente. Esses resultados demonstram que a hierarquia foi eficaz em discriminar ataques distribuídos de negação de serviço frente às demais categorias de intrusão.

Por fim, as amostras corretamente classificadas como não DDoS no terceiro nível são encaminhadas ao quarto nível da hierarquia, destinado à diferenciação entre as demais categorias de ataque (*DoS*, *Mirai*, *Spoofing*, *Web-Based* e *Brute Force*). A Figura 18 apresenta a matriz de confusão obtida nessa última fase, e a Tabela 18 consolida os resultados por classe original.

A matriz evidencia que o classificador alcançou desempenho elevado para as classes majoritárias. No caso de ataques *DoS*, 1.158.966 instâncias foram corretamente identificadas, com apenas 20 amostras distribuídas de forma incorreta entre outras classes. Para a

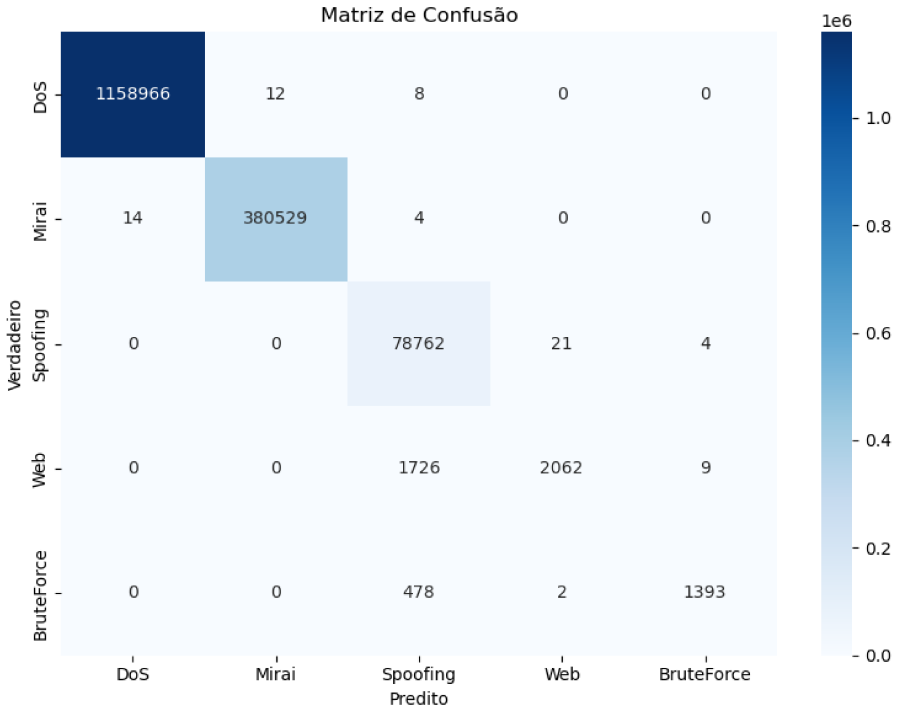

Figura 18 – Matriz de confusão do modelo Multi-Ataque BRDM.

categoria *Mirai*, 380.529 das 380.547 amostras foram classificadas corretamente, restando apenas 18 erros. O mesmo padrão se repete na classe *Spoofing*, que obteve 78.762 acertos de um total de 78.787 instâncias, resultando em apenas 25 erros residuais.

Nas classes minoritárias, nota-se uma maior sensibilidade a erros de classificação. A classe *Web*, composta por 3.797 instâncias, apresentou 2.062 amostras corretamente identificadas, mas registrou confusões consideráveis com outras classes, sobretudo com *Spoofing* (1.726 casos). De maneira semelhante, a categoria *Brute Force*, composta por 1.873 amostras, registrou 1.393 acertos, mas contabilizou 480 erros, sendo 478 deles decorrentes de confusões com *Spoofing*.

De forma geral, os resultados indicam que o modelo Multiataque do BRDM apresenta desempenho robusto na identificação das classes volumosas, como *DoS*, *Mirai* e *Spoofing*. Contudo, o desempenho cai de forma perceptível em classes menos representativas, especialmente *Web* e *Brute Force*, que concentram a maior parte das confusões. Esse resultado reflete diretamente o desbalanceamento do conjunto de dados, revelando que, embora o classificador seja altamente eficaz para ataques predominantes, há margem para aprimoramentos no tratamento de categorias minoritárias.

Comparando os resultados obtidos pelos classificadores Multi-Ataque do BDM e do BRDM, observa-se que ambos apresentaram desempenho consistente nas classes majoritárias, especialmente em *DoS* e *Mirai*, atingindo altas taxas de acerto. Entretanto, enquanto o classificador BDM apresentou maior dificuldade na detecção da categoria *Spoofing*, com

77.895 acertos, o BRDM elevou esse valor para 78.762 acertos, evidenciando que a etapa de filtragem de ataques *Recon* contribuiu para reduzir as confusões dessa classe. Em ambos os casos, as classes minoritárias, *Web-Based* e *Brute Force*, permaneceram como as mais desafiadoras, refletindo o impacto do desbalanceamento do conjunto de dados e resultando em quedas perceptíveis de desempenho. Ainda assim, o BRDM demonstrou ganhos relevantes também nessas categorias. Assim, embora os dois modelos demonstrem elevada eficácia na detecção das classes predominantes, o BRDM se destaca por mitigar parte das confusões tanto em classes intermediárias quanto minoritárias, enquanto o BDM revela maior fragilidade nesses cenários.

Para sustentar essa análise, a Tabela 18 contabiliza a média de acertos e erros de cada categoria para o modelo multi-ataque ao longo de dez execuções.

Tabela 18 – Análise das classes originais para o nível 4 do classificador BRDM.

	Total de Registros	**Acertos**	**Erros**	**Recall**
DoS	1158993	1158970	23	0,9999
Mirai	380548	380533	15	0,9999
Spoofing	78621	78592	29	0,9996
BruteForce	1857	1382	475	0,7442
Web	3799	2103	1696	0,5535

Analisando a tabela, a classe *DoS*, composta por 1.158.993 instâncias, apresentou 1.158.970 acertos e apenas 23 erros, resultando em um *recall* de 99,99%. De forma semelhante, a classe *Mirai*, com 380.548 amostras, obteve 380.533 classificações corretas e 15 incorretas, também atingindo um *recall* de 99,99%. Esses valores demonstram que o classificador é altamente eficaz na identificação de ataques volumosos e recorrentes, refletindo sua capacidade de generalização frente a padrões majoritários.

A classe *Spoofing*, embora menor que as anteriores, também apresentou desempenho robusto, com 78.592 acertos em 78.621 instâncias e apenas 29 erros, alcançando um *recall* de 99,96%. Esse resultado indica que o modelo mantém bom nível de discriminação mesmo em categorias intermediárias, ainda que se observe um leve aumento no número relativo de erros em comparação às classes mais numerosas.

Em contrapartida, as classes minoritárias demonstraram desempenho significativamente inferior. A categoria *Brute Force*, composta por 1.857 amostras, obteve 1.382 acertos e 475 erros, com *recall* de apenas 74,42%. Já a classe *Web*, com 3.799 instâncias, apresentou apenas 2.103 classificações corretas e 1.696 equivocadas, resultando em um *recall* de 55,35%. Esses valores evidenciam que a hierarquia proposta, embora contribua para reduzir ambiguidades em classes intermediárias, ainda não é suficiente para mitigar totalmente os efeitos do desbalanceamento extremo das categorias minoritárias.

Contabilizando o total de erros e acertos em cada um dos níveis, durante as dez execuções, foi possível montar uma tabela para o classificador BRDM como um todo,

analisando o seu desempenho por classe. Esses resultados são mostrados na Tabela 19.

Tabela 19 – Análise das classes alvo do classificador BRDM.

	Total de Registros	**Acertos**	**Erros**	**Recall**
Mirai	380553	380533	20	0,9999
DDoS	3162103	3161825	278	0,9999
DoS	1159198	1158970	228	0,9998
Benigno	197672	190828	6844	0,9653
Spoofing	87563	78592	8971	0,8975
Recon	63760	54016	9744	0,8471
BruteForce	2352	1382	970	0,5875
Web	4469	2103	2366	0,4705

A análise consolidada das classes originais para o classificador BRDM evidencia um comportamento heterogêneo entre categorias majoritárias e minoritárias. As classes *Mirai*, *DDoS* e *DoS*, que concentram a maior parte das instâncias no conjunto de dados, apresentaram um excelente desempenho, com *recall* de 99,99% e 99,98%, confirmando a robustez do modelo frente a ataques volumétricos e recorrentes. O tráfego *Benigno*, embora volumoso, apresentou taxa de acerto inferior (96,53%), com mais de 6.800 instâncias mal classificadas como maliciosas, evidenciando certa dificuldade do classificador em evitar falsos positivos. Nas classes intermediárias, como *Spoofing* (89,75%) e *Recon* (84,71%), nota-se desempenho satisfatório, mas ainda limitado em relação às categorias majoritárias, indicando maior suscetibilidade a confusões. Já as categorias minoritárias, *Brute Force* (58,75%) e *Web-Based* (47,05%), apresentaram os menores valores de *recall*, com quase metade das instâncias sendo incorretamente atribuídas a outras classes. Esses resultados evidenciam que, embora o BRDM seja altamente eficaz para detectar classes predominantes, seu desempenho é significativamente impactado pelo desbalanceamento do conjunto de dados.

Comparando os resultados consolidados dos classificadores BDM e BRDM, observa-se que ambos alcançaram desempenho notável nas classes majoritárias, com *recall* próximo de 100% para *DDoS*, *DoS* e *Mirai*, confirmando a robustez de ambas as abordagens frente a esses tipos de ataques. O tráfego benigno apresentou desempenho equivalente em ambos os modelos, com *recall* de 96,53%, resultado coerente com o fato de que o processo de treinamento aplicado aos dois classificadores foi o mesmo. No entanto, observam-se diferenças pontuais no tratamento das categorias intermediárias, ainda que os valores obtidos sejam próximos. O BRDM apresentou desempenho superior ao BDM na detecção de ataques do tipo *Spoofing* (89,75% contra 88,80%), sugerindo uma melhora na capacidade de reduzir confusões por meio da filtragem hierárquica. Por outro lado, o BRDM obteve desempenho ligeiramente inferior em relação à classe *Recon* (84,71% contra 85,63%), evidenciando que a introdução da hierarquia não necessariamente beneficia todas as categorias de maneira uniforme. Nas classes minoritárias, ambos os classificadores obtiveram desempenhos

abaixo do desejável, mas o BRDM apresentou avanços importantes, com *Brute Force* atingindo 58,75% contra 51,36% no BDM e *Web-Based* alcançando 47,05% frente a 43,52%. Dessa forma, embora as duas arquiteturas sejam eficazes na detecção das classes predominantes, o BRDM mostra-se mais vantajoso por mitigar parte das limitações observadas em categorias menos representativas, ainda que o impacto do desbalanceamento continue influenciando significativamente o desempenho global.

Os resultados obtidos pelo BRDM também evidenciam que esse modelo conseguiu integrar as vantagens dos classificadores binários e multiclasse, alcançando um equilíbrio entre eficácia na filtragem de ataques e capacidade de distinguir múltiplos tipos de intrusão. A introdução de um nível específico para a identificação de ataques do tipo *Recon* contribuiu para mitigar confusões recorrentes no BDM, em especial nas classes *Spoofing*, *Brute Force* e *Web-Based*. Dessa forma, a abordagem hierárquica não apenas preserva a robustez observada nas classes majoritárias, como também promove ganhos relevantes em categorias mais desafiadoras. Portanto, o BRDM mostra-se uma alternativa promissora para reduzir sobreposições entre classes e aprimorar a detecção de ataques em cenários caracterizados pela alta complexidade e desbalanceamento dos dados.

4.4.3 Avaliação Computacional

Para complementar a análise do classificador BRDM, foi realizada uma avaliação em termos de desempenho computacional. A Tabela 20 apresenta os resultados médios obtidos para cada um dos modelos que compõem a hierarquia, considerando dez execuções independentes.

Tabela 20 – Resultado das métricas de desempenho computacional para os modelos que compõem o classificador BRDM.

	Benigno	**Recon**	**DDoS**	**Multi-Ataque**
Tempo de Treino	39,11 min	49,63 min	45,34 min	14,52 min
Tempo de Teste	0,005 ms	0,005 ms	0,006 ms	0,006 ms
Tamanho	123.464 bytes	123.464 bytes	123.464 bytes	125.920 bytes

Com relação ao desempenho computacional dos modelos isolados, observa-se que o tempo de treinamento mais elevado ocorreu no modelo *Recon*, com média de 49,63 minutos, seguido pelo modelo *DDoS*, com 45,34 minutos. Já o modelo Benigno, responsável pela separação entre tráfego legítimo e malicioso, apresentou tempo de treinamento próximo ao observado no BDM (39,11 minutos). O modelo multi-ataque, apesar de lidar com múltiplas classes, foi consideravelmente mais rápido para treinar, demandando apenas 14,52 minutos para ser construído, possivelmente em razão de a quantidade de registros envolvidos ser significativamente menor que nos modelos anteriores. No que diz respeito ao tempo de inferência, todos os modelos apresentaram valores bastante reduzidos, variando entre 0,005 e 0,006 milissegundos por instância. Quanto ao tamanho em memória,

os modelos que utilizam a abordagem binária ocuparam um espaço de 123.464 bytes, enquanto o modelo multi-ataque ocupou 125.920 bytes.

Para fins de simplificação, o desempenho computacional do classificador BRDM como sistema integrado foi estimado pela soma dos custos de seus quatro níveis hierárquicos, desconsiderando o eventual *overhead* associado à comunicação entre modelos. O tempo total de treinamento foi de aproximadamente 149 minutos, superior ao observado no BDM (109 minutos) e também aos classificadores planos, que demandaram 39 minutos no binário e 64 minutos no multiclasse. Esse aumento é restrito à fase de construção do modelo e, portanto, não compromete sua utilização em produção. Durante a inferência, o tempo médio de teste por amostra foi de 22 microssegundos, maior que no BDM (19 microssegundos), mas ainda adequado para cenários de detecção em tempo real. Em termos de memória, o BRDM apresentou tamanho total aproximado de 496.312 bytes, também superior ao BDM (373.624 bytes) e, consequentemente, aos classificadores planos.

Esse crescimento em tempo e uso de memória é esperado, uma vez que o BRDM incorpora quatro modelos na hierarquia, um a mais que o BDM. Apesar disso, o custo adicional é compensado pela proposta de maior especialização na detecção de ataques de reconhecimento, buscando reduzir confusões identificadas na etapa anterior. Assim, o BRDM apresenta-se como uma solução computacionalmente mais exigente, mas que busca entregar ganhos qualitativos em termos de classificação hierárquica.

4.4.4 Experimento com XGBoost

Tendo em vista que o BRDM se sobressaiu em relação às demais abordagens testadas, esta pesquisa buscou investigar a mesma abordagem hierárquica substituindo, em cada um dos níveis, o algoritmo de classificação *Random Forest* pelo *Extreme Gradient Boosting* (*XGBoost*). A escolha desse algoritmo fundamenta-se em estudos recentes que apontam o *XGBoost* como uma alternativa promissora para problemas de classificação utilizando o conjunto de dados CICIoT2023 (ESTEVES, 2025).

O processo de construção do BRDM foi mantido, alterando-se apenas o algoritmo empregado em cada nível da hierarquia. Os experimentos foram conduzidos no mesmo ambiente computacional, preservando-se as divisões de treino e teste em todas as dez execuções independentes, de modo a garantir condições justas de comparação.

Entretanto, o desempenho preditivo obtido com essa nova configuração mostrou-se inferior ao observado com o uso do *Random Forest*. Por esse motivo, a análise dos resultados será apresentada de forma mais sucinta, contemplando: (i) o desempenho individual dos modelos que compõem a hierarquia, (ii) a tabela consolidada de erros e acertos para cada classe no classificador integrado e (iii) a avaliação do desempenho computacional.

Primeiramente, para a avaliação individual de cada modelo, foram calculadas as

métricas tradicionais de desempenho médio ao longo das dez execuções. Os resultados estão sintetizados na Tabela 21.

Tabela 21 – Resultado das métricas de desempenho preditivo para os modelos que compõem o classificador BRDM XGBoost.

	Benigno	**Recon**	**DDoS**	**Multi-Ataque**
Acurácia	0,9952	0,9979	0,9998	0,9984
Precisão	0,9618	0,9589	0,9998	0,9435
Recall	0,9766	0,9509	0,9997	0,7898
F1	0,9691	0,9549	0,9997	0,8432

A análise dos resultados médios do classificador BRDM XGBoost evidencia um comportamento heterogêneo entre os diferentes níveis da hierarquia. O modelo binário inicial apresentou desempenho consistente, com acurácia de 99,52% e valores equilibrados de precisão (96,18%) e *recall* (97,66%), resultando em um F1 de 96,91%, indicando um bom desempenho geral para este modelo. No segundo nível, o modelo Recon obteve acurácia ligeiramente superior (99,79%), mas apresentou valores mais modestos de precisão (95,89%) e *recall* (95,09%), refletindo maior suscetibilidade a confusões nessa categoria, ainda que mantendo desempenho satisfatório (F1 de 95,49%).

Já o modelo *DDoS* destacou-se como o nível mais robusto da hierarquia, alcançando alta acurácia (99,98%) e métricas extremamente elevadas em todas as dimensões (precisão de 99,98%, *recall* de 99,97% e F1 de 99,97%). Em contrapartida, o modelo Multi-Ataque apresentou a maior fragilidade, apesar da alta acurácia (99,84%). A precisão manteve-se elevada (94,35%), indicando baixo número de falsos positivos, mas o *recall* caiu para 78,98%, refletindo uma quantidade significativa de falsos negativos e um F1 de apenas 84,32%. Esse resultado mostra que, embora o modelo consiga rejeitar corretamente instâncias negativas, tem maior dificuldade em detectar todas as ocorrências das classes minoritárias, comprometendo a sensibilidade global da hierarquia nesse nível. De forma geral, os resultados sugerem que o BRDM *XGBoost* manteve alta eficácia para classes mais volumosas, mas sofreu perda considerável de desempenho na fase de classificação multiataque, especialmente em termos de *recall*.

Em seguida, para possibilitar uma avaliação mais aprofundada, os experimentos contabilizaram o total de erros e acertos por categoria em cada um dos níveis, durante as dez execuções. Dessa forma, foi possível montar uma tabela para o classificador BRDM *XGBoost* contemplando o total de instâncias, número de acertos, número de erros e valor de *recall* obtido. Esses resultados são mostrados na Tabela 22.

Analisando a tabela, percebe-se que as classes *Mirai*, *DDoS* e *DoS* apresentaram um excelente desempenho, com valores de *recall* próximos a 100%. O tráfego *Benigno* apresentou taxa de acerto de 95,63%, que apesar de bom, evidencia certa dificuldade do modelo. Nas classes intermediárias, como *Spoofing* (87,82%) e *Recon* (81,07%), nota-

Tabela 22 – Análise das classes alvo do classificador BRDM XGBoost.

	Total de Registros	**Acertos**	**Erros**	**Recall**
Mirai	380553	380527	26	0,9999
DDoS	3162103	3161619	484	0,9998
DoS	1159198	1158667	531	0,9995
Benigno	197672	189050	8622	0,9563
Spoofing	87563	76905	10658	0,8782
Recon	63760	51695	12065	0,8107
Web	4469	1274	3195	0,2850
BruteForce	2352	512	1840	0,2176

se desempenho satisfatório, mas com maior suscetibilidade a confusões. Já as categorias minoritárias, *Brute Force* (21,76%) e *Web-Based* (28,50%), apresentaram desempenho significativamente inferior, indicando que o modelo enfrenta dificuldades em detectar corretamente ataques com menor representatividade no conjunto de dados. Essa baixa taxa de *recall* sugere que o modelo tende a confundir essas classes com outras mais frequentes, evidenciando um possível viés de classificação devido ao desbalanceamento de classes.

Comparando os resultados consolidados dos classificadores BRDM *XGBoost* e BRDM *Random Forest*, observa-se que ambos alcançaram desempenho semelhante nas classes *DDoS*, *DoS*, *Mirai* e Benigno. Contudo, o *Random Foret* obteve resultados significativamente melhores para as classes *Spoofing* (89,75% contra 87,82%) e *Recon* (84,71% contra 81,07%). Essa diferença entre os classificadores foi ainda mais crítica para as categorias *Brute Force* (58,75% contra 21,76%) e *Web-Based* (47,05% contra 28,50%).

Por fim, para completar a análise do classificador BRDM XGBoost, foi feita uma avaliação em termos de desempenho computacional. A Tabela 23 apresenta os resultados obtidos por cada um dos modelos, considerando uma média de dez execuções independentes.

Tabela 23 – Resultado das métricas de desempenho computacional para os modelos que compõem o classificador BRDM XGBoost.

	Benigno	**Recon**	**DDoS**	**Multi-Ataque**
Tempo de Treino	60,43 s	54,33 s	45,81 s	72,33 s
Tempo de Teste	0,0003 ms	0,0001 ms	0,0001 ms	0,002 ms
Tamanho	5.328 bytes	5.312 bytes	5.296 bytes	5.344 bytes

O desempenho computacional do BRDM *XGBoost* apresentou resultados bastante favoráveis em termos de tempo de execução e tamanho dos modelos. Os tempos médios de treino foram relativamente baixos, variando de 45,81 segundos no modelo *DDoS* até 72,33 segundos no modelo Multi-Ataque, o que demonstra eficiência no processo de treinamento mesmo considerando a complexidade da hierarquia. Durante a fase de teste, o tempo de inferência foi significativamente baixo em todos os casos, com valores variando entre 0,0001

e 0,002 milissegundos, o que torna a abordagem altamente adequada para cenários que demandam respostas em tempo real. Além disso, o tamanho final dos modelos foi em torno de 5 KB cada, evidenciando o baixo consumo de memória e a leveza computacional dessa solução.

Quando comparado ao BRDM *Random Forest*, as diferenças são expressivas. Enquanto o *XGBoost* necessitou de segundos para treinar cada modelo, o *Random Forest* demandou dezenas de minutos, variando de 14,52 minutos (Multi-Ataque) até 49,63 minutos (*Recon*), o que representa uma diferença significativa em termos de tempo de treinamento. Na fase de teste, ambos apresentaram tempos bastante reduzidos, mas o *XGBoost* mais uma vez foi superior ao *Random Forest*. Em relação ao tamanho dos modelos, o contraste é evidente: os modelos do *Random Forest* ultrapassaram 120 KB, enquanto os do *XGBoost* permaneceram próximos a 5 KB. Dessa forma, ainda que o desempenho preditivo do *Random Forest* seja mais consistente, o *XGBoost* se destaca amplamente em eficiência computacional, apresentando-se como uma alternativa mais leve e escalável para ambientes com restrições de tempo e recursos.

4.5 Considerações e Lições Aprendidas

Esta seção apresenta uma síntese dos principais resultados obtidos nos experimentos, destacando as considerações relevantes e as lições aprendidas a partir da avaliação dos classificadores planos e hierárquicos para detecção de intrusões em redes IoT. A Tabela 24 resume a comparação entre os experimentos, evidenciando o objetivo central de cada classificador, bem como suas respectivas vantagens e limitações.

Tabela 24 – Comparação entre os experimentos.

	Objetivo Principal	Vantagens Principais	Limitações Principais
Modelo Binário	Distinguir entre tráfego benigno e malicioso em geral	- Alta sensibilidade na detecção de ataques (poucos falsos negativos)	- Perda de informações sobre o tipo específico de ataque
Modelo Multiclasse	Distinguir entre 8 categorias de tráfego (benigno e 7 tipos de ataque)	- Granularidade na identificação de tipos de ataques	- Maior complexidade e risco de erros com o aumento do número de classes - Maior taxa de falsos negativos para ataques em geral - Dificuldade e baixa performance para classes minoritárias (Web-Based, Brute Force, Recon e Spoofing)
Modelo BDM	Combinar a eficiência da filtragem binária com a granularidade da multiclasse	- Combina a alta sensibilidade do binário com a capacidade de categorização do multiclasse - Melhora a interpretabilidade e a especialização das etapas de classificação - Desempenho superior ao multiclasse plano nas classes Recon e Spoofing	- Custo computacional total significativamente maior que classificadores planos - Desempenho limitado e confusões persistentes em classes minoritárias devido ao desbalanceamento - Desempenho inferior ao classificador multiclasse para as categorias Benigno, Brute Force e Web-Based
Modelo BRDM (Random Forest)	Estender o BDM, adicionando um nível específico para ataques de Reconhecimento, visando mitigar confusões e melhorar a precisão geral	- Combina a alta sensibilidade do binário com a capacidade de categorização do multiclasse - Melhora a interpretabilidade e a especialização das etapas de classificação - Desempenho superior nas classes Spoofing, Brute Force e Web-Based, em comparação ao BDM e ao multiclasse	- Custo computacional total ainda mais elevado que o BDM - Desempenho ainda impactado pelo desbalanceamento em categorias minoritárias - Desempenho ligeiramente inferior para a classe Recon em comparação ao BDM
Modelo BRDM (XGBoost)	Investigar a abordagem BRDM utilizando o algoritmo XGBoost para comparação de desempenho e eficiência computacional	- Excelente eficiência computacional	- Desempenho preditivo inferior ao BRDM Random Forest, especialmente no modelo Multi-Ataque - Maior suscetibilidade a confusões em classes intermediárias e minoritárias

A análise dos classificadores planos revelou um claro *trade-off* entre o desempenho preditivo e a granularidade da detecção. O classificador binário destacou-se pela elevada sensibilidade na identificação de ataques, reduzindo a probabilidade de falsos negativos, o que é crucial em cenários de cibersegurança. No entanto, não forneceu informações detalhadas sobre o tipo específico de ataque. Já o classificador multiclasse, embora com menor desempenho na detecção de ataques, ofereceu a capacidade de distinguir entre diferentes tipos de ataque, proporcionando informações valiosas para estratégias de mitigação direcionadas. Em termos de eficiência computacional, o classificador binário mostrou-se mais eficiente, com menor tempo de treinamento (aproximadamente 39 minutos) e de teste por amostra (0,005 ms), além de um tamanho em memória ligeiramente menor (123.464 bytes) em comparação ao multiclasse (cerca de 64 minutos de treino, 0,011 ms de teste e 128.344 bytes), embora ambos tenham sido considerados viáveis para execução.

Para combinar as vantagens e mitigar as desvantagens dos classificadores planos, propôs-se o classificador hierárquico BDM (Benigno, *DDoS* e Multi-Ataque), que explorou a hierarquia dos ciberataques. O modelo Benigno, no primeiro nível, herdou a alta sensibilidade do classificador binário na identificação de ataques. O modelo *DDoS*, no segundo nível, apresentou desempenho excepcional devido à natureza volumétrica e alta representatividade desses ataques no conjunto de dados, facilitando sua separação e mitigando o desbalanceamento. O modelo Multi-Ataque, no terceiro nível, conseguiu desempenho similar ao *multiclasse* plano na distinção entre tipos de ataques restantes, com ganhos pontuais em *Recon* e *Spoofing*. Contudo, as classes minoritárias *Brute Force* e *Web-Based* continuaram sendo as mais desafiadoras, evidenciando o impacto do desbalanceamento de classes. O BDM apresentou um tempo médio de treinamento total de aproximadamente 109 minutos e um tamanho de aproximadamente 373.624 bytes, superiores aos classificadores planos, mas com um tempo de teste por amostra (19 microssegundos) ainda adequado para aplicações em tempo real.

Dando sequência à investigação, o classificador hierárquico BRDM (Benigno, *Recon*, *DDoS* e Multi-Ataque) foi desenvolvido para abordar as confusões com ataques de reconhecimento observadas no BDM, adicionando um nível específico para a detecção de ataques *Recon*. O modelo Benigno manteve a consistência na filtragem inicial. A inclusão do modelo *Recon* no segundo nível demonstrou boa capacidade de separar outros ataques de ataques de reconhecimento, contribuindo para reduzir ambiguidades, embora seu desempenho na detecção de *Recon* tenha sido ligeiramente inferior ao do modelo Multi-Ataque do BDM. Os modelos *DDoS* e Multi-Ataque subsequentes se beneficiaram da filtragem prévia de *Recon*, mostrando desempenho superior em classes intermediárias (*Spoofing*) e minoritárias (*Brute Force* e *Web-Based*) em comparação ao BDM. O BRDM demonstrou ser uma alternativa promissora para reduzir sobreposições entre classes e aprimorar a detecção de ataques em cenários complexos e desbalanceados. Computacionalmente, o BRDM apresentou um tempo total de treinamento de aproximadamente 149

minutos e um tamanho total de aproximadamente 496.312 bytes, sendo mais exigente que o BDM e os classificadores planos. No entanto, o tempo de teste por amostra (22 microssegundos) permaneceu aceitável para aplicações em tempo real.

Um experimento adicional com o BRDM, substituindo o algoritmo *Random Forest* por *XGBoost*, revelou um *trade-off* notável entre desempenho preditivo e eficiência computacional. O desempenho preditivo geral do BRDM *XGBoost* foi inferior ao BRDM *Random Forest*, especialmente no modelo Multi-Ataque. Isso sugere que o *Random Forest* foi mais adequado para lidar com a heterogeneidade e o desbalanceamento da classificação multiataque. Em contrapartida, a eficiência computacional do *XGBoost* foi amplamente superior, com tempos de treinamento medidos em segundos contra minutos para *Random Forest*, tempos de teste ainda menores (0,0001 a 0,002 ms) e modelos significativamente mais leves (aproximadamente 5 KB cada contra mais de 120 KB para *Random Forest*). O *XGBoost* mostrou-se uma alternativa mais escalável para ambientes com restrições de recursos.

A partir dos experimentos realizados, observou-se que a classificação hierárquica é uma abordagem promissora para a detecção de intrusões em redes *IoT*. Ela combina alta sensibilidade na identificação geral de ataques com a capacidade de caracterizar diferentes tipos de intrusão, superando as limitações dos classificadores planos. A definição da hierarquia desempenha um papel crucial: a inclusão de níveis intermediários voltados a classes com características específicas - como ataques volumétricos (*DDoS*) e preparatórios (*Recon*) - contribui para refinar a detecção, reduzir ambiguidades e simplificar os classificadores subsequentes, resultando em ganhos de desempenho.

Entretanto, o desbalanceamento de classes permanece um desafio relevante, mesmo em arquiteturas hierárquicas, prejudicando a detecção de categorias minoritárias. Além disso, a escolha do algoritmo de aprendizado de máquina deve ser cuidadosamente considerada, buscando equilibrar desempenho preditivo e eficiência computacional.

Por fim, embora classificadores hierárquicos demandem maior uso de recursos (tempo de treinamento e memória) em função da sua complexidade, o tempo de inferência mantém-se suficientemente baixo para a maioria das aplicações em tempo real, o que os torna viáveis em diversos cenários práticos.

5 Conclusão

Este trabalho de conclusão de curso teve como objetivo principal construir um classificador hierárquico para detecção de intrusões em redes IoT, capaz de combinar as qualidades de ambos os classificadores binários e planos. A motivação para este estudo reside na crescente proliferação de dispositivos IoT e no consequente aumento das vulnerabilidades cibernéticas, tornando essencial o desenvolvimento de Sistemas de Detecção de Intrusão (IDS) mais eficazes para esses ambientes. A abordagem hierárquica surgiu como uma alternativa promissora para superar as limitações dos classificadores planos tradicionais, combinando a eficiência da filtragem binária de ataques com a granularidade da classificação multiclasse.

A metodologia adotada consistiu na extração e pré-processamento de uma subamostra estratificada do conjunto de dados CICIoT2023, seguida pela construção e avaliação de classificadores planos (binário e multiclasse) e hierárquicos (BDM e BRDM). Todos os modelos foram implementados utilizando o algoritmo *Random Forest*, com um experimento adicional para o BRDM utilizando *XGBoost*. A validação do desempenho preditivo dos modelos propostos foi realizada sistematicamente utilizando o conjunto de dados CICIoT2023. O desempenho preditivo foi mensurado por métricas como acurácia, precisão, *recall* e *F1-Score*, e o desempenho computacional foi avaliado por tempo de treinamento, tempo de teste e uso de memória RAM.

Diante dos resultados obtidos, conclui-se que o objetivo principal deste trabalho foi alcançado. Os classificadores hierárquicos propostos demonstraram ser capazes de combinar a alta sensibilidade dos classificadores binários na detecção de ataques com a capacidade de diferenciação dos classificadores multiclasse, superando as limitações dos classificadores planos e validando a hipótese de que a organização hierárquica constitui uma alternativa eficaz para a detecção de intrusões em redes IoT.

Os classificadores planos revelaram um *trade-off* notável: o classificador binário destacou-se pela alta sensibilidade na detecção de ataques, crucial para evitar falsos negativos em cibersegurança, mas sem granularidade. Em contrapartida, o classificador multiclasse ofereceu a capacidade de distinguir entre diferentes tipos de ataque, sendo eficaz na detecção de tráfego benigno, mas inferior na detecção de ataques.

Para endereçar essas limitações, foi desenvolvido o classificador hierárquico BDM (Benigno, *DDoS* e Multi-Ataque). Ele conseguiu combinar a alta sensibilidade do classificador binário na detecção de ataques (99,75% de acertos) com a granularidade do classificador multiclasse na distinção dos tipos de intrusão, atendendo ao objetivo central desta pesquisa. O modelo Benigno manteve o mesmo desempenho do classificador binário

na detecção de tráfego legítimo e malicioso. O modelo *DDoS*, no segundo nível, apresentou desempenho considerável (*recall* de 99,99%) devido à sua natureza volumétrica e alta representatividade, reduzindo a complexidade do nível subsequente. Por fim, o modelo Multi-Ataque, no terceiro nível, obteve resultados comparáveis ao multiclasse plano, com ganhos pontuais em *Recon* (+2%) e *Spoofing* (+1,7%), e perdas em *Brute Force* (-3,3%) e *Web-Based* (-0,1%, valor praticamente desprezível).

A avaliação do classificador BDM possibilitou a análise dos erros em cada nível hierárquico, permitindo identificar as classes mais problemáticas e os tipos de ataques que ainda representam desafios. As principais confusões ocorreram entre classes minoritárias, como *Web-Based* e *Brute Force*, que foram frequentemente classificadas como *Recon* ou *Spoofing*. Além disso, embora a taxa de detecção dessas duas últimas tenha aumentado, observou-se uma considerável sobreposição entre seus padrões, resultando em confusões recorrentes entre *Recon* e *Spoofing*.

Ao investigar diferentes formas de organização hierárquica com o objetivo de reduzir as confusões observadas no classificador BDM e aprimorar a detecção de ataques, foi desenvolvido um segundo classificador hierárquico: o BRDM (Benigno, *Recon*, *DDoS* e Multi-Ataque). Essa arquitetura buscou mitigar, em especial, os erros associados aos ataques de reconhecimento. O modelo Benigno manteve o mesmo desempenho do classificador binário na detecção de tráfego legítimo e malicioso. A introdução de um nível específico para a classe *Recon* (modelo Recon) demonstrou boa capacidade de identificação desses ataques (recall de 92,16%). Entretanto, seu principal benefício foi a redução das confusões de outras classes com *Recon*, alcançando recall de 99,93% e contribuindo para diminuir ambiguidades na classificação. Os modelos subsequentes - *DDoS* e Multi-Ataque - se beneficiaram dessa filtragem prévia, apresentando desempenho superior em relação ao BDM. Foram observados ganhos relevantes nas classes *Spoofing* (+1%), *Brute Force* (+7,4%) e *Web-Based* (+3,5%), ultrapassando também o classificador multiclasses.

Dessa forma, o BRDM demonstrou ser uma alternativa promissora para reduzir as sobreposições entre classes e aprimorar a detecção de ataques em cenários complexos e desbalanceados. O modelo não apenas alcançou a capacidade de distinção de tipos de intrusão do classificador multiclasse, como também a superou em algumas categorias específicas, como *Brute Force* e *Web-Based* (+3,5%).

Entretanto, as principais confusões permaneceram concentradas nas classes minoritárias. Em particular, *Brute Force* apresentou recall de 58,75% e *Web-Based* de 47,05%, o que evidencia que quase metade de suas instâncias foi incorretamente atribuída a outras classes. Além disso, *Spoofing* (89,75%) e *Recon* (84,71%) também mostraram certa suscetibilidade a erros, embora tenham mantido um desempenho considerado satisfatório.

Sob a perspectiva computacional, o classificador binário demonstrou ser o mais eficiente, com um tempo médio de treinamento de aproximadamente 39 minutos, um

tempo de teste por amostra de 0,005 ms e um tamanho de 123.464 bytes em memória RAM. O classificador multiclasse, por sua vez, foi ligeiramente mais exigente, registrando cerca de 64 minutos de treinamento, 0,011 ms por amostra no teste e 128.344 bytes de memória. Já as abordagens hierárquicas apresentaram custos computacionais superiores: o classificador BDM teve um tempo médio de treinamento total de aproximadamente 109 minutos e um tamanho de cerca de 373.624 bytes, com tempo de teste de 19 microssegundos por amostra. O BRDM, sendo o mais complexo com quatro níveis, demandou cerca de 149 minutos de treinamento e um tamanho de aproximadamente 496.312 bytes, com um tempo de teste de 22 microssegundos por amostra.

Um experimento adicional com o BRDM, substituindo o algoritmo *Random Forest* por *XGBoost*, revelou um *trade-off* notável entre desempenho preditivo e eficiência computacional. O desempenho preditivo geral do BRDM *XGBoost* foi inferior ao BRDM *Random Forest*, especialmente no modelo Multi-Ataque. Isso sugere que o *Random Forest* foi mais adequado para lidar com a heterogeneidade e o desbalanceamento da classificação multiataque. Em contrapartida, a eficiência computacional do *XGBoost* foi amplamente superior, com tempos de treinamento medidos em segundos (variando de 45,81s a 72,33s), tempos de teste ainda menores (0,0001 a 0,002 ms) e modelos significativamente mais leves (aproximadamente 5 KB contra mais de 120 KB para *Random Forest*). O *XGBoost* mostrou-se uma alternativa mais escalável para ambientes com restrições de recursos.

A definição da hierarquia mostrou-se um aspecto central: a inclusão de níveis intermediários dedicados a classes com características específicas, como *DDoS* e *Recon*, contribui para refinar a detecção e reduzir ambiguidades. Além disso, a análise detalhada dos erros de classificação por categoria revelou-se fundamental para identificar classes problemáticas e orientar o aprimoramento contínuo dos sistemas.

A partir das análises realizadas, observa-se que os classificadores hierárquicos são particularmente indicados para cenários em que há forte desbalanceamento entre classes, elevada heterogeneidade dos ataques e necessidade simultânea de alta sensibilidade na detecção de tráfego malicioso e caracterização detalhada dos tipos de intrusão. Em contrapartida, em cenários com restrições severas de recursos computacionais ou em aplicações onde apenas a detecção binária de ataques é suficiente, classificadores planos podem representar uma solução mais adequada.

Em suma, este estudo evidenciou, que a classificação hierárquica constitui uma abordagem viável e eficaz para a detecção de intrusões em redes IoT, superando o desempenho de classificadores planos em cenários que demandam um equilíbrio entre alta sensibilidade na identificação de ataques, capacidade de diferenciação entre classes e interpretabilidade dos resultados. Nesse contexto, os modelos BDM e BRDM mostraram-se capazes de integrar a robustez da detecção binária com a granularidade da classificação multiclasse, fornecendo subsídios mais precisos para a construção de estratégias de defesa

cibernética direcionadas.

Para trabalhos futuros, diversas vertentes de pesquisa podem ser exploradas para aprimorar ainda mais a detecção de intrusões em redes IoT usando classificação hierárquica. Primeiramente, é crucial desenvolver e aplicar técnicas para mitigar o desbalanceamento de classes, um desafio persistente que impactou negativamente o desempenho dos classificadores hierárquicos, especialmente nas categorias minoritárias como *Web-Based* e *Brute Force*. Além disso, dado que os classificadores hierárquicos, embora eficazes, demandam mais recursos computacionais do que as abordagens planas, há uma oportunidade para otimizar a eficiência computacional desses modelos, tanto em tempo de treinamento e teste quanto em uso de memória, possivelmente explorando outras configurações de algoritmos ou arquiteturas que demonstrem um melhor *trade-off* entre desempenho preditivo e requisitos de recursos, como sugerido pelo experimento com *XGBoost*.

Adicionalmente, pode-se investigar diferentes formas de organização da hierarquia de classificadores, buscando estruturas ainda mais refinadas que explorem as características diferenciais dos ataques para reduzir a confusão entre classes e otimizar a detecção, assim como o BRDM aprimorou o BDM. Por fim, a utilização de estratégias de classificação semi-supervisionada no primeiro nível da hierarquia surge como uma vertente promissora, especialmente no contexto do Modelo Benign. Essa abordagem permitiria explorar grandes volumes de dados de tráfego em redes IoT - geralmente não rotulados - para fortalecer a capacidade de distinção inicial entre tráfego legítimo e malicioso. Com isso, seria possível reduzir a dependência de bases totalmente rotuladas, que costumam ser limitadas ou onerosas de obter em cenários reais.

Referências

AGGARWAL, C. C. **Neural Networks and Deep Learning: A Textbook**. [S.l.]: Springer, 2023. Citado na página 23.

_____. **Neural networks and deep learning: A textbook**. [S.l.]: Springer International Publishing Springer, 2023. Citado na página 23.

AHMAD, R.; ALSMADI, I.; ALHAMDANI, W.; TAWALBEH, L. A comprehensive deep learning benchmark for iot ids. **Computers & Security**, v. 114, p. 102588, 2022. ISSN 0167-4048. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167404821004119>. Citado na página 19.

ALANI, M. M.; DAMIANI, E. Xrecon: An explainbale iot reconnaissance attack detection system based on ensemble learning. **Sensors**, v. 23, n. 11, 2023. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/23/11/5298>. Citado 2 vezes nas páginas 17 e 18.

ALBANDES, R.; YAMIN, A.; BARBOSA, J.; MACHADO, R. Abordagem I3VSM: Acompanhamento autônomo de pacientes explorando ciência de situação na IoT. In: **Anais de XXXVII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais**. [S.l.]: Sociedade Brasileira de Telecomunicações, 2019. Citado na página 17.

ALIN, F.; CHEMCHEM, A.; NOLOT, F.; FLAUZAC, O.; KRAJECKI, M. Towards a hierarchical deep learning approach for intrusion detection. In: **Machine Learning for Networking**. Cham: Springer International Publishing, 2020, (Lecture notes in computer science). p. 15–27. Citado na página 30.

ARDABILI, S. F.; MOSAVI, A.; GHAMISI, P.; FERDINAND, F.; VARKONYI-KOCZY, A. R.; REUTER, U.; RABCZUK, T.; ATKINSON, P. M. Covid-19 outbreak prediction with machine learning. **Algorithms**, MDPI, v. 13, n. 10, p. 249, 2020. Citado na página 19.

BAHGA, A.; MADISETTI, V. **Internet of things: A hand-on approach**. [S.l.]: Universities Press (India), 2015. Citado na página 16.

BELCIC, I.; STRYKER, C. **O que É Aprendizado Supervisionado?** 2025. Acessado em 8 de julho de 2025. Disponível em: <https://www.ibm.com/br-pt/think/topics/supervised-learning>. Citado na página 20.

BERGAMASCO, D. A.; GASETA, E. R.; CAETANO, L. C. et al. **Entendendo IPS e IDS: uso de IPS e IDS em conjunto**. [S.l.]: 004, 2021. <https://ric.cps.sp.gov.br/handle/123456789/12805>. Monografia. Acessado em 2 de maio de 2025. Citado na página 18.

BIAU, G.; SCORNET, E. A random forest guided tour. **Test**, Springer, v. 25, n. 2, p. 197–227, 2016. Citado na página 23.

BISHOP, C. M. **Pattern Recognition and Machine Learning**. [S.l.]: Springer, 2006. Citado na página 20.

CHEN, T.; HE, T.; BENESTY, M.; KHOTILOVICH, V.; TANG, Y.; CHO, H.; CHEN, K.; MITCHELL, R.; CANO, I.; ZHOU, T. et al. Xgboost: extreme gradient boosting. **R package version 0.4-2**, v. 1, n. 4, p. 1–4, 2015. Citado na página 23.

CNN Brasil. **Ataques hackers aumentam 8,8% no Brasil e país segue como 2º mais atacado do mundo**. 2024. <https://www.cnnbrasil.com.br/nacional/ataques-hackers-aumentam-88-no-brasil-e-pais-segue-como-2o-mais-atacado-do-mundo/>. Acessado em 25 de abril de 2025. Citado na página 13.

_____. **Brasil é vice-campeão em ataques cibernéticos, com 1.379 golpes por minuto, aponta estudo | CNN Brasil — cnnbrasil.com.br**. 2024. <https://www.cnnbrasil.com.br/economia/negocios/brasil-e-vice-campeao-em-ataques-ciberneticos-com-1-379-golpes-por-minuto-aponta-estudo/>. [Accessed 25-04-2025]. Citado na página 12.

DESAI, M.; SHAH, M. An anatomization on breast cancer detection and diagnosis employing multi-layer perceptron neural network (mlp) and convolutional neural network (cnn). **Clinical eHealth**, Elsevier, v. 4, p. 1–11, 2021. Citado na página 19.

ESTEVES, J. P. R. **Aplicação de Inteligência Artificial Explicável no Contexto de Detecção de Intrusão em Dispositivos IoT**. 54 p. Monografia (Trabalho de Conclusão de Curso (Graduação em Ciência da Computação)) — Universidade Federal de Uberlândia, Uberlândia, 2025. Citado 4 vezes nas páginas 13, 28, 37 e 79.

EURICO, I. C. N. **Estratégias de monitorização de tráfego em cidades inteligentes**. Dissertação (Mestrado) — Universidade do Minho (Portugal), 2023. Citado na página 17.

FACELI, K.; LORENA, A. C.; GAMA, J.; CARVALHO, A. C. P. d. L. F. d. **Inteligência artificial: uma abordagem de aprendizado de máquina**. [S.l.]: LTC, 2011. Citado na página 20.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. The kdd process for extracting useful knowledge from volumes of data. **Commun. ACM**, Association for Computing Machinery, New York, NY, USA, v. 39, n. 11, p. 27–34, nov. 1996. ISSN 0001-0782. Disponível em: <https://doi.org/10.1145/240455.240464>. Citado na página 20.

FERNANDAURIAS. **5 desafios E 5 tendências Emergentes Em Cibersegurança**. 2023. Disponível em: <https://www.cnnbrasil.com.br/branded-content/tecnologia/por-dell-technologies-5-desafios-e-5-tendencias-emergentes-em-ciberseguranca/>. Nenhuma citação no texto.

FIRMINO, L. **Passado, presente e Futuro da Segurança Cibernética**. 2024. Disponível em: <https://securityleaders.com.br/passado-presente-e-futuro-da-seguranca-cibernetica/#:~:text=A%20hist%C3%B3ria%20da%20ciberseguran%C3%A7a%20remonta,por%20empresas%2C%20governos%20e%20indiv%C3%ADduos.> Nenhuma citação no texto.

GHASSEMI, M.; NAUMANN, T.; SCHULAM, P.; BEAM, A. L.; CHEN, I. Y.; RANGANATH, R. A review of challenges and opportunities in machine learning for health. **AMIA Summits on Translational Science Proceedings**, v. 2020, p. 191, 2020. Citado na página 19.

HAFEZIAN, Z.; NADERAN, M.; JADERYAN, M. A machine learning-based approach for multi-class intrusion detection and classification in iot using ciciot2023 dataset. In: **2024 11th International Symposium on Telecommunications (IST)**. [S.l.: s.n.], 2024. p. 161–165. Citado 3 vezes nas páginas 11, 13 e 28.

HASAN, M.; ISLAM, M. M.; ZARIF, M. I. I.; HASHEM, M. Attack and anomaly detection in iot sensors in iot sites using machine learning approaches. **Internet of Things**, v. 7, p. 100059, 2019. ISSN 2542-6605. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2542660519300241>. Citado na página 26.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. [S.l.]: Springer, 2009. Citado na página 22.

HORCHULHACK, P.; SANTIN, A. O.; VIEGAS, E. K. Atualização confiável dos modelos de detecção de intrusão baseada em aprendizagem de máquina. In: SBC. **Concurso de Teses e Dissertações (CTD)**. [S.l.], 2024. p. 98–107. Citado na página 19.

IoT Analytics. **State of IoT 2024: Number of connected IoT devices growing 13% to 18.8 billion globally**. 2025. <https://iot-analytics.com/number-connected-iot-devices/>. IoT Analytics. Acessado em: 03 set. 2025. Citado na página 13.

JOO, D.; HONG, T.; HAN, I. The neural network models for ids based on the asymmetric costs of false negative errors and false positive errors. **Expert Systems with Applications**, v. 25, n. 1, p. 69–75, 2003. ISSN 0957-4174. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0957417403000071>. Citado na página 11.

KUBAT, M. **An Introduction to Machine Learning**. 2. ed. Cham, Switzerland: Springer International Publishing, 2017. Citado na página 24.

KUMAR, A. G.; RASTOGI, A.; RANGA, V. Evaluation of different machine learning classifiers on new iot dataset ciciot2023. In: **2024 International Conference on Intelligent Systems for Cybersecurity (ISCS)**. [S.l.: s.n.], 2024. p. 1–6. Citado 3 vezes nas páginas 13, 27 e 28.

LI, Y.; LIU, Q. A comprehensive review study of cyber-attacks and cyber security; emerging trends and recent developments. **Energy Rep.**, Elsevier BV, v. 7, p. 8176–8186, nov. 2021. Citado na página 18.

LORENA, A. C.; CARVALHO, A. C. P. L. F. de; GAMA, J. M. P. A review on the combination of binary classifiers in multiclass problems. **Artif. Intell. Rev.**, Springer Science and Business Media LLC, v. 30, n. 1-4, p. 19–37, dez. 2008. Citado na página 29.

MADAKAM, S.; RAMASWAMY, R.; TRIPATHI, S. Internet of things (IoT): A literature review. **J. Comput. Commun.**, Scientific Research Publishing, Inc,, v. 03, n. 05, p. 164–173, 2015. Citado na página 17.

MARIANO, D. Métricas de avaliação em machine learning: acurácia, sensibilidade, precisão, especificidade e f-score. In: **BIOINFO - Revista Brasileira de Bioinformática e Biologia Computacional**. [S.l.]: Alfahelix, 2021. Citado na página 24.

MASEER, Z. K.; YUSOF, R.; BAHAMAN, N.; MOSTAFA, S. A.; FOOZY, C. F. M. Benchmarking of machine learning for anomaly based intrusion detection systems in the cicids2017 dataset. **IEEE Access**, v. 9, p. 22351–22370, 2021. Citado 2 vezes nas páginas 19 e 26.

MCKINNEY, W. Data structures for statistical computing in python. In: **Proceedings of the Python in Science Conference**. [S.l.]: SciPy, 2010. p. 56–61. Citado na página 14.

MITCHELL, T. M. **Machine Learning**. [S.l.]: McGraw-Hill, 1997. Citado na página 19.

MOHD, N.; SINGH, A.; BHADAURIA, H. S. Intrusion detection system based on hybrid hierarchical classifiers. **Wirel. Pers. Commun.**, Springer Science and Business Media LLC, v. 121, n. 1, p. 659–686, nov. 2021. Citado na página 31.

MOSAFI, F. **Machine Learning: metodologia de mineração automatizada com dados das redes sociais e processamento de linguagem natural**. [S.l.]: Editora Dialética, 2022. Citado na página 20.

MRCET. **Intrusion Detection Systems**. [S.l.], 2025. Acessado em: 24 mar. 2025. Disponível em: <https://mrcet.com/pdf/Lab%20Manuals/CSECS/INTRUSION%20DETECTION%20SYSTEMS.pdf>. Citado na página 19.

NARAYAN, K. G. R.; MOOKHERJI, S.; ODELU, V.; PRASATH, R.; TURLAPATY, A. C.; DAS, A. K. IIDS: Design of intelligent intrusion detection system for Internet-of-Things applications. 2023. Citado na página 34.

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>. Citado 7 vezes nas páginas 12, 17, 18, 19, 27, 34 e 35.

NTAMWIZA, J. M. V.; BWIRE, H. Predicting biking preferences in kigali city: A comparative study of traditional statistical models and ensemble machine learning models. **Transport Economics and Management**, v. 3, p. 78–91, 2025. ISSN 2949-8996. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2949899625000048>. Citado 2 vezes nas páginas 22 e 23.

Pandas Developers. **pandas**. 2025. <https://pandas.pydata.org/>. Online; acessado em 17 de agosto de 2025. Citado 2 vezes nas páginas 36 e 52.

PH.D., J. M.; KAVLAKOGLU, E. **What are classification models?** 2025. [Accessed 10-08-2025]. Disponível em: <https://www.ibm.com/think/topics/classification-models>. Citado na página 21.

PIETRO, R. D.; MANCINI, L. V. Approaches in anomaly-based network intrusion dete. In: **Intrusion Detection Systems**. Boston, MA: Springer US, 2008, (Advances in information security). p. 1–16. Citado na página 19.

Pympler Developers. **Pympler: Development and Documentation**. 2024. <https://pympler.readthedocs.io/en/stable/index.html>. Acessado em: 01 set. 2025. Citado na página 51.

QUINLAN, J. R. Learning decision tree classifiers. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 28, n. 1, p. 71–72, 1996. Citado na página 23.

SARIKAYA, A.; KILIÇ, B. G. A class-specific intrusion detection model: Hierarchical multi-class IDS model. **SN Comput. Sci.**, Springer Science and Business Media LLC, v. 1, n. 4, jul. 2020. Citado na página 30.

SARNOVSKY, M.; PARALIC, J. Hierarchical intrusion detection using machine learning and knowledge model. **Symmetry (Basel)**, MDPI AG, v. 12, n. 2, p. 203, fev. 2020. Citado na página 30.

Scikit-learn Developers. **Scikit-learn: Machine Learning in Python**. 2025. <https://scikit-learn.org/stable/>. Acessado em 25 de abril de 2025. Citado 4 vezes nas páginas 14, 36, 51 e 52.

SHALEV-SHWARTZ, S.; BEN-DAVID, S. **Understanding machine learning**. [S.l.]: Cambridge University Press, 2014. Citado 2 vezes nas páginas 19 e 21.

SHANMUGAM, V.; RAZAVI-FAR, R.; HALLAJI, E. Addressing class imbalance in intrusion detection: A comprehensive evaluation of machine learning approaches. **Electronics**, v. 14, n. 1, 2025. ISSN 2079-9292. Disponível em: <https://www.mdpi.com/2079-9292/14/1/69>. Citado na página 35.

SILLA JR, C. N.; FREITAS, A. A. A survey of hierarchical classification across different application domains. **Data Min. Knowl. Discov.**, Springer Science and Business Media LLC, v. 22, n. 1-2, p. 31–72, jan. 2011. Citado 2 vezes nas páginas 21 e 29.

STEINBERG, J.; BEAVER, K.; WINKLER, I.; COOMBS, T. **Cybersecurity all-in-one for dummies**. [S.l.]: For Dummies, 2023. Citado na página 17.

THOMAS, K.; BENJAMIN, R.-K.; FERNANDO, P.; BRIAN, G.; MATTHIAS, B.; JONATHAN, F.; KYLE, K.; JESSICA, H.; JASON, G.; SYLVAIN, C.; PAUL, I.; DAMIÁN, A.; SAFIA, A.; CAROL, W.; Jupyter Development Team. Jupyter notebooks – a publishing format for reproducible computational workflows. In: **Positioning and Power in Academic Publishing: Players, Agents and Agendas**. [S.l.]: IOS Press, 2016. Citado 2 vezes nas páginas 14 e 53.

UDDIN, M. A.; ARYAL, S.; BOUADJENEK, M. R.; AL-HAWAWREH, M.; TALUKDER, M. A. Hierarchical classification for intrusion detection system: Effective design and empirical analysis. **Ad Hoc Netw.**, Elsevier BV, v. 178, n. 103982, p. 103982, nov. 2025. Citado 3 vezes nas páginas 12, 31 e 32.

XGBoost Developers. **XGBoost Documentation**. 2022. <https://xgboost.readthedocs.io/en/stable/index.html>. Online; acessado em 17 de março de 2025. Citado na página 14.

ZARGAR, S. T.; JOSHI, J.; TIPPER, D. A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks. **IEEE Communications Surveys & Tutorials**, v. 15, n. 4, p. 2046–2069, 2013. Citado na página 18.

ZARPELÃO, B. B.; MIANI, R. S.; KAWAKANI, C. T.; ALVARENGA, S. C. D. A survey of intrusion detection in internet of things. **Journal of Network and Computer Applications**, Elsevier, v. 84, p. 25–37, 2017. Citado na página 18.