UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Vitor Yuji Federico Matsushita

Análise Comparativa de Modelos de Aprendizado de Máquina para Detecção de Ataques DDoS em Ambientes IoT com Seleção de *features* Baseada em SHAP

Uberlândia, Brasil

2026

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Vitor Yuji Federico Matsushita

Análise Comparativa de Modelos de Aprendizado de Máquina para Detecção de Ataques DDoS em Ambientes IoT com Seleção de *features* Baseada em SHAP

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: Diego Nunes Molinos

Universidade Federal de Uberlândia – UFU

Faculdade de Computação

Uberlândia, Brasil

2026

Vitor Yuji Federico Matsushita

Análise Comparativa de Modelos de Aprendizado de Máquina para Detecção de Ataques DDoS em Ambientes IoT com Seleção de *features* Baseada em SHAP

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, 07 de Agosto de 2026:

Diego Nunes Molinos
Orientador

Fernanda Maria da Cunha Santos
Professor

Igor da Penha Natal
Professor

Uberlândia, Brasil
2026

Agradecimentos

Agradeço a minha família, a minha namorada, a meu orientador, Diego Nunes Molinos que sempre me apoiou; e a todos que contribuíram de alguma forma para a elaboração deste trabalho.

Resumo

Este trabalho realiza um estudo comparativo sobre os efeitos da seleção de atributos orientada por interpretabilidade SHAP (*SHapley Additive exPlanations*) no desempenho de classificadores supervisionados de aprendizado de máquina aplicados à detecção de ataques de Negação de Serviço Distribuída (DDoS) em ambientes de Internet das Coisas (IoT). O interesse nesse cenário decorre das restrições de capacidade de processamento, memória e da grande heterogeneidade de dispositivos característicos desse tipo de rede, fatores que dificultam a adoção direta de soluções de segurança tradicionais. Foram avaliados cinco algoritmos de aprendizado supervisionado – Árvore de Decisão, Floresta Aleatória (*Random Forest*), Máquina de Vetores de Suporte (SVM), AdaBoost e Perceptron de Múltiplas Camadas (MLP) – treinados sobre um subconjunto balanceado do *dataset* CICIoT2023, submetido a um processo de higienização numérica e balanceamento amostral prévio. A técnica SHAP foi empregada não apenas como ferramenta de interpretabilidade *post-hoc*, mas como critério de consenso entre os cinco modelos para ranquear e selecionar subconjuntos reduzidos de características (Top 5, Top 10 e Top 15), permitindo comparar o desempenho de cada classificador em função do número de atributos utilizados. Os resultados experimentais indicam que o subconjunto de 10 características representa um ponto de equilíbrio entre desempenho preditivo e redução da dimensionalidade do problema para a maioria dos modelos avaliados, com a Floresta Aleatória apresentando o melhor desempenho global (acurácia de 99,80%) com variação inferior a 0,1 ponto percentual mesmo nos subconjuntos reduzidos. Os resultados evidenciam, ainda, a sensibilidade de classificadores baseados em margens geométricas (SVM) à ausência de normalização das variáveis de entrada, em contraste com a robustez observada nos modelos baseados em árvore. Conclui-se que o consenso entre valores SHAP de múltiplos modelos constitui uma estratégia viável de seleção de atributos para problemas de classificação de tráfego de rede, contribuindo para a discussão sobre o uso de técnicas de interpretabilidade como apoio à engenharia de atributos em cenários de recursos computacionais restritos.

Palavras-chave: DDoS. IoT. Edge Computing. Seleção de *features*. Aprendizado de Máquina. SHAP.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ANN	Redes Neurais Artificiais (*Artificial Neural Networks*)
AUC	Área Sob a Curva (*Area Under the Curve*)
BMI	*Broadcast Music, Inc.*
CIC	*Canadian Institute for Cybersecurity*
CNN	Redes Neurais Convolucionais (*Convolutional Neural Networks*)
DDoS	Negação de Serviço Distribuída (*Distributed Denial of Service*)
DL	Aprendizado Profundo (*Deep Learning*)
EDR	*Endpoint Detection and Response*
EUA	Estados Unidos da América
FBI	*Federal Bureau of Investigation*
HTTP	*Hypertext Transfer Protocol*
IAT	Tempo Entre Chegadas (*Inter-Arrival Time*)
IDS	Sistema de Detecção de Intrusão (*Intrusion Detection System*)
IoMT	Internet das Coisas Médicas (*Internet of Medical Things*)
IoT	Internet das Coisas (*Internet of Things*)
IPS	Sistema de Prevenção de Intrusão (*Intrusion Prevention System*)
LIME	*Local Interpretable Model-Agnostic Explanations*
LSTM	Memória de Longo Prazo (*Long Short-Term Memory*)
M2M	Comunicação Máquina a Máquina (*Machine-to-Machine*)
MitM	*Man-in-the-Middle*
ML	Aprendizado de Máquina (*Machine Learning*)
MLP	Perceptron de Múltiplas Camadas (*Multilayer Perceptron*)
MPAA	*Motion Picture Association of America*

NDR	*Network Detection and Response*
NIST	Instituto Nacional de Padrões e Tecnologia (*National Institute of Standards and Technology*)
NTA	Análise de Tráfego de Rede (*Network Traffic Analysis*)
OSI	*Open Systems Interconnection*
OTA	Atualização Remota (*Over-The-Air*)
PCA	Análise de Componentes Principais (*Principal Component Analysis*)
RIAA	*Recording Industry Association of America*
ROC	*Receiver Operating Characteristic*
RSSF	Redes de Sensores Sem Fio
SHAP	*SHapley Additive exPlanations*
SOPA	*Stop Online Piracy Act*
SVM	Máquina de Vetores de Suporte (*Support Vector Machine*)
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*
WAF	*Web Application Firewall*
XAI	Inteligência Artificial Explicável (*Explainable AI*)

Lista de símbolos

FN_i	Quantidade de Falsos Negativos referentes à classe i
FP_i	Quantidade de Falsos Positivos referentes à classe i
k	Número total de classes ou agrupamentos analisados
$O(j)$	Número de ocorrências da *feature* j nos modelos validados
PC_n	n-ésimo Componente Principal extraído via PCA
$S_{m,j}$	Importância absoluta da *feature* j conferida pelo modelo m
$\bar{S}(j)$	Magnitude média global da importância SHAP para a *feature* j
TFP_i	Taxa de Falsos Positivos (*False Positive Rate*) para a classe i
Top_m	Subconjunto das 15 *features* de maior importância do modelo m
TVP_i	Taxa de Verdadeiros Positivos (*True Positive Rate*) para a classe i
VN_i	Quantidade de Verdadeiros Negativos referentes à classe i
VP_i	Quantidade de Verdadeiros Positivos referentes à classe i
$x_{m,j}$	Variável binária de presença da *feature* j no Top 15 do modelo m
χ^2	Teste estatístico Qui-Quadrado

Sumário

1 Introdução

Ambientes de Internet das Coisas (IoT), que englobam ecossistemas inteligentes, polos industriais automatizados (*Smart Factories*), automação residencial e escritórios conectados, têm-se consolidado como infraestrutura crítica da sociedade contemporânea, principalmente com o advento da Inteligência Artificial (KUCHUK; MALOKHVII, 2024). A IoT oferece uma vasta gama de serviços contínuos pela internet, viabilizando a coleta e a troca contínua de dados entre sensores, atuadores, sistemas embarcados e dispositivos móveis. Contudo, a maior parte desses terminais opera na borda da rede (*Edge Computing*), com severas restrições de processamento, armazenamento e consumo de energia. Com a expansão acelerada dessas redes e a comunicação por canais abertos e sem fio, as vulnerabilidades a invasões cibernéticas crescem significativamente (FILHO, 2020). Atacantes exploram essas brechas para interceptar, eliminar ou modificar pacotes, visando a exfiltração de dados sensíveis ou a apropriação de privilégios de usuários legítimos.

Devido à limitação computacional inerente ao hardware de IoT, a implementação de medidas de defesa tradicionais é um desafio. As interfaces de programação de aplicações (APIs) desses dispositivos frequentemente carecem de autenticação robusta, e o ciclo de vida do *firmware* raramente inclui atualizações de segurança regulares. (SONAR; UPADHYAY, 2014) Sob esta óptica, usuários maliciosos aproveitam essas falhas para enviar códigos maliciosos (*payloads*) e assumir o controle do sistema operacional embarcado. Uma vez comprometidos, centenas de milhares de dispositivos vulneráveis são transformados em aparelhos “zumbis"e incorporados a redes massivas controladas por um atacante, conhecidas como (*botnets*) (Palo Alto Networks, 2024).

Em virtude desse comprometimento, essa infraestrutura distribuída é mobilizada para orquestrar ataques de Negação de Serviço Distribuída (DDoS, do inglês *Distributed Denial of Service*) em escala global. Ao executar técnicas como a inundação de pacotes (*packet flooding*) e a exaustão das tabelas de conexão, os atacantes comprometem severamente a disponibilidade de servidores e serviços-alvo, inviabilizando operações críticas (SONAR; UPADHYAY, 2014).

O alto impacto dos ataques DDoS e a progressiva fragilidade do ecossistema IoT são ressaltados por indicadores da indústria. A Oracle, em seu estudo técnico (Oracle, 2020), destacou a existência de mais de 7 bilhões de dispositivos inteligentes conectados globalmente, projetando a superação da marca de 22 bilhões de terminais ativos em 2025. Corroborando a gravidade desse avanço, a *Check Point Software Technologies* registrou, em 2023, um salto expressivo de 41% no número médio de ataques semanais direcionados especificamente a dispositivos IoT em comparação com o ano anterior, incidindo com

maior severidade sobre organizações europeias, seguidas pelos polos tecnológicos da Ásia e da América Latina (Check Point Research, 2023).

Segundo (DONNO et al., 2018), um dos principais exemplos de ataques DDoS explorando vulnerabilidades na IoT ocorreu no final de 2016, com o surgimento do *malware Mirai*. Ao explorar credenciais padrão de fábrica em câmeras IP e roteadores domésticos, o Mirai infectou centenas de milhares de terminais e, em 21 de outubro de 2016, disparou um ataque que gerou um volume de tráfego massivo de aproximadamente 1,2 terabits por segundo. A agressão resultou na indisponibilidade em massa de provedores de infraestrutura de DNS e de serviços digitais de alcance mundial, como Twitter, Netflix e Spotify, o que mudou definitivamente a percepção global sobre o risco inerente à computação distribuída.

De acordo com (Palo Alto Networks, 2024), em janeiro de 2012, o cibergrupo hacktivista *Anonymous* mobilizou ofensivas DDoS coordenadas em protesto contra o projeto de lei *Stop Online Privacy Act* (SOPA). A operação desativou simultaneamente os portais governamentais do Departamento de Justiça dos Estados Unidos, do *Federal Bureau of Investigation* (FBI) e da Casa Branca, bem como os domínios das corporações de mídia *Motion Picture Association of America* (MPAA), *Recording Industry Association of America* (RIAA) e *Universal Music Group*.

1.1 Justificativa

Mecanismos de defesa tradicionais baseados em assinaturas estáticas dependem de regras pré-compiladas de ameaças conhecidas (OLIVEIRA, 2023). Contudo, essa abordagem revela-se ineficaz no dinamismo do ecossistema IoT, pois é incapaz de reconhecer ataques de dia zero (*zero-day*) e demonstra falta de sensibilidade a pequenas variações nos pacotes maliciosos. Para superar essa condição, abordagens baseadas em Aprendizado de Máquina (ML, do inglês *Machine Learning*) vêm sendo utilizadas para compor Sistemas de Detecção de Intrusão (IDS, do inglês *Intrusion Detection Systems*) adaptativos e baseados em comportamento (OLIVEIRA, 2023). No entanto, a aplicação prática de Inteligência Artificial diretamente na borda da rede (*Edge Computing*) enfrenta dois obstáculos metodológicos:

1. **Dimensionalidade e Custo Computacional:** Sistemas modernos de captura de rede extraem dezenas de métricas estatísticas a partir de cada fluxo. Processar muitas variáveis ao mesmo tempo em roteadores ou microcontroladores em tempo real consome ciclos de processamento e largura de banda proibitivos, além de induzir ruído que degrada a precisão de classificadores profundos.

2. **Capacidade Preditiva e Explicabilidade:** Algoritmos complexos comportam-se

como “caixas-pretas", impedindo que engenheiros auditem o raciocínio da detecção. Paralelamente, modelos puramente não supervisionados (como detectores de anomalias) falham em cenários de inundações DDoS, pois a repetição massiva de pacotes hostis gera um falso padrão de densidade que cega o algoritmo, levando-o a confundir o ataque com o tráfego regular.

Para solucionar essa lacuna, este Trabalho de Conclusão de Curso propõe, utilizando o conjunto de dados de referência moderno **CICIoT2023**, uma investigação que adota a técnica de interpretabilidade *SHapley Additive exPlanations* (SHAP) como filtro para ranquear e selecionar subconjuntos enxutos de características (Top 5, Top 10 e Top 15). Ao analisar o comportamento dos algoritmos nesses cenários, o estudo identifica o ponto ótimo entre a eficácia preditiva e a viabilidade computacional em dispositivos com recursos limitados.

Utilizando o conjunto de dados de referência moderna (NETO et al., 2023), a pesquisa investiga o impacto direto da Engenharia e Seleção de *features* (*Feature Selection*) sobre múltiplos classificadores supervisionados (Árvore de Decisão, Floresta Aleatória, AdaBoost, Redes Neurais MLP e SVM).

Diferentemente da literatura tradicional, que utiliza o SHAP apenas como ferramenta de visualização *post-hoc*, este trabalho o emprega como um **filtro ativo de Engenharia de *features* (*Feature Selection*)**. Ao quantificar a contribuição exata de cada variável de rede, o pipeline proposto reduz a dimensionalidade do conjunto de dados para subconjuntos enxutos (Top 5, Top 10 e Top 15 características), viabilizando o processamento em hardware restrito. A avaliação empírica demonstra que classificadores de *ensemble*, como a Floresta Aleatória (*Random Forest*), mantêm um teto de acurácia global de 99,80% mesmo operando sobre o espaço de *features* drasticamente reduzido, garantindo alta eficácia defensiva com latência de inferência mínima.

1.2 Objetivos do trabalho

Diante da crescente vulnerabilidade dos ambientes conectados e das severas restrições computacionais dos dispositivos de borda, este trabalho tem como objetivo principal desenvolver um estudo comparativo sobre modelos de Aprendizado de Máquina (ML) para a detecção de ataques de Negação de Serviço Distribuída (DDoS) no ecossistema da Internet das Coisas (IoT) com foco na análise do consenso de *features*. Para viabilizar a classificação de tráfego em tempo real na borda da rede (*Edge Computing*) com alto desempenho preditivo e reduzida carga computacional, este trabalho investiga o impacto direto da Engenharia e Seleção de *features* (*Feature Selection*) utilizando a metodologia de interpretabilidade SHAP como filtro ativo. A abordagem visa quantificar empiricamente em que medida diferentes cortes dimensionais (subconjuntos das Top 5, Top 10 e Top

15 características) influenciam a acurácia, a estabilidade e a viabilidade operacional de múltiplos classificadores.

Para alcançar o objetivo geral, estabelecem-se os seguintes objetivos específicos:

- Realizar a revisão da literatura com foco em mapear as principais vulnerabilidades de segurança desses ambientes e analisar a mecânica, a evolução e os padrões volumétricos dos ataques DDoS alavancados por *botnets*;
- Projetar um *pipeline* sequencial de pré-processamento, higienização, balanceamento amostral e supressão de multicolinearidade (Filtro de Pearson), garantindo a integridade matemática dos dados submetidos aos classificadores;
- Treinar, validar e comparar o desempenho de múltiplos classificadores supervisionados (*Random Forest*, Árvore de Decisão, *AdaBoost*, Redes Neurais Artificiais e SVM) em cenários experimentais;
- Aplicar a interpretabilidade do SHAP para realização de consenso e importância preditiva de variáveis, estabelecendo cortes dimensionais progressivos (subconjuntos das Top 5, Top 10 e Top 15 características mais relevantes);

1.3 Questões de Pesquisa

Com base no objetivo geral e específicos, duas questões de pesquisa foram formuladas, a saber:

RQ1: Qual é o impacto da seleção de diferentes subconjuntos consensuados de *features*: Top 5, Top 10 e Top 15, definidos por meio do SHAP, no desempenho dos modelos de Aprendizado de Máquina para a detecção de ataques DDoS em ambientes IoT?

RQ2: Qual dos classificadores supervisionados avaliados apresenta o melhor equilíbrio entre desempenho preditivo e custo no que tange o volume de *features*, ao utilizar conjuntos reduzidos de *features* do CICIoT2023?

Essas questões estão alinhadas ao foco do trabalho: a primeira investiga o efeito da redução de dimensionalidade, sendo a segunda compara os modelos e busca identificar a solução mais adequada para dispositivos de borda com recursos limitados.

1.4 Organização da Monografia

Este documento está estruturado em oito capítulos, organizados da seguinte forma:

- **Capítulo 1 — Introdução:** Apresenta o contexto do cenário IoT, a justificativa, o problema de pesquisa, os objetivos e a organização do texto.
- **Capítulo 2 — Fundamentação Teórica:** Descreve a arquitetura e os protocolos da IoT, a mecânica dos ataques DDoS, o uso de Aprendizado de Máquina em Sistemas de Detecção de Intrusão (IDS) e o método SHAP.
- **Capítulo 3 — Trabalhos Correlatos:** Analisa os estudos recentes sobre detecção de intrusão e explicabilidade, comparando as soluções existentes com a abordagem proposta neste trabalho.
- **Capítulo 4 — Método:** Detalha a base de dados CICIoT2023, as etapas de pré-processamento, as configurações dos classificadores e o processo de seleção de *features* via SHAP.
- **Capítulo 5 — Experimentos:** Apresenta a execução dos testes, os resultados do cenário de referência (*baseline*) e a aplicação do SHAP para o ranqueamento das variáveis.
- **Capítulo 6 — Análise de Resultados:** Discute e compara o desempenho dos cinco classificadores nos diferentes cenários (espaço integral, Top 5, Top 10 e Top 15), avaliando o impacto da redução de dimensionalidade.
- **Capítulo 7 — Conclusão:** Sintetiza os resultados obtidos e as contribuições práticas da união entre florestas de decisão e interpretabilidade para a segurança na borda da rede.
- **Capítulo 8 — Trabalhos Futuros:** Sugere os próximos passos para a pesquisa, como a implementação dos modelos em *hardware* físico de borda e testes adicionais de normalização de dados.
- **Capítulo 8 — Código-Fonte dos Experimentos:** Referência para o código fonte dos experimentos.

2 Fundamentação Teórica

Este capítulo apresenta a base teórica necessária para o entendimento da pesquisa. Inicialmente, descreve-se o ecossistema da Internet das Coisas (IoT), passando por sua arquitetura, protocolos e limitações de *hardware*. Na sequência, são discutidas as principais ameaças a esses ambientes, com foco nos ataques de Negação de Serviço Distribuída (DDoS). Por fim, o texto detalha as técnicas de Análise de Tráfego de Rede (NTA) e os Sistemas de Detecção de Intrusão (IDS) baseados em Aprendizado de Máquina (ML), destacando a necessidade de otimizar esses algoritmos para a borda da rede e o uso do método SHAP para a interpretabilidade e seleção de *features*.

2.1 O Ecossistema da Internet das Coisas (IoT)

A Internet das Coisas (IoT, do inglês *Internet of Things*) é uma rede que conecta objetos físicos à Internet, permitindo a troca de dados por meio de sensores e atuadores usando protocolos padronizados. Seu objetivo prático é monitorar e gerenciar ativos de forma autônoma, expandindo a comunicação tradicional da internet para interações máquina a máquina (M2M, do inglês *Machine-to-Machine*).

O termo *Internet of Things* foi criado por Kevin Ashton em 1999, focado inicialmente no uso de etiquetas de radiofrequência (RFID). O conceito, no entanto, demorou alguns anos para amadurecer. Até 2005, os estudos concentravam-se principalmente em Redes de Sensores Sem Fio (RSSF), que ajudaram a desenvolver técnicas para lidar com restrições de energia, memória e escalabilidade. Foi apenas entre 2008 e 2010, com o avanço das RSSFs e das redes móveis, que a IoT passou a ser adotada em larga escala (SANTOS et al., 2023).

2.1.1 Arquitetura e Plataformas

A estrutura de um sistema IoT é tradicionalmente modelada em camadas abstratas que definem o fluxo dos dados desde a captura física até a entrega ao usuário final. O modelo referencial mais difundido divide-se em três camadas essenciais:

- **Camada de Percepção (ou Física):** Composta pelos sensores, atuadores e microcontroladores de borda responsáveis por interagir com o mundo real, coletando parâmetros físicos (como temperatura, movimento e posicionamento) ou executando ações mecânicas.

- **Camada de Rede:** Encarregada da transmissão, roteamento e processamento inicial dos dados coletados. Utiliza protocolos de comunicação sem fio e cabeados para interligar os terminais de borda à Internet ou a servidores de processamento em nuvem.

- **Camada de Aplicação:** Nível em que os dados são processados, analisados e estruturados para serem utilizados em serviços inteligentes ao usuário final, como painéis de monitoramento de cidades inteligentes ou automação hospitalar (AL-FUQAHA et al., 2015).

2.1.2 Protocolos de Comunicação

A conectividade na camada de rede da IoT depende de protocolos leves, adaptados para operar em cenários com largura de banda restrita, alta latência e limitações energéticas severas. Entre os padrões mais adotados (SONAR; UPADHYAY, 2014).Destacam-se:

- **MQTT (*Message Queuing Telemetry Transport*):** Protocolo de mensageria assíncrono baseado no modelo de publicação/assinatura (*publish/subscribe*), ideal para comunicações M2M com baixo consumo de cabeçalho e alta confiabilidade em redes instáveis.

- **CoAP (*Constrained Application Protocol*):** Desenvolvido pela *Internet Engineering Task Force* (IETF) para operar sobre o protocolo UDP (*User Datagram Protocol*), atua como uma alternativa leve ao HTTP em dispositivos restritos, traduzindo o modelo de transferência web para arquiteturas de baixa potência.

- **Protocolos de Curto Alcance:** Tecnologias como ***Bluetooth Low Energy* (BLE)** e **Zigbee** são amplamente implementadas na criação de Redes de Área Pessoal (*Personal Area Networks* (PANs), permitindo a comunicação entre nós próximos com mínimo consumo de bateria (BIBI et al., 2021).

2.1.3 Aplicações e Dispositivos de Borda

A IoT é aplicada em diversos setores. Um exemplo prático é a Internet das Coisas Médicas (IoMT, do inglês *Internet of Medical Things*), que integra dispositivos biomédicos aos sistemas de saúde para monitorar parâmetros vitais em tempo real. Esse cenário exige conexões de baixa latência e controles rigorosos de privacidade dos dados (AL-NBHANY; ZAHARY; AL-SHARGABI, 2024). Além disso, sensores ambientais (temperatura, pressão e umidade) e de posicionamento (como acelerômetros e GPS) formam a base da automação industrial e dos dispositivos vestíveis (*wearables*).

Para conectar esses sensores físicos na camada de percepção, utilizam-se sistemas embarcados e módulos de comunicação compactos. Entre os hardwares mais comuns nessa arquitetura estão:

- **ESP-01 (família ESP8266/ESP32):** Módulos Wi-Fi de baixo custo e baixo consumo de energia. São amplamente utilizados na indústria e na comunidade para integrar conectividade sem fio a microcontroladores e sensores de forma nativa.
- **Arduino:** Plataforma de prototipagem eletrônica de código aberto (*open-source*). Baseada em microcontroladores simples, é muito utilizada para a leitura de sensores e automação básica devido à sua facilidade de programação e integração rápida (AL-NBHANY; ZAHARY; AL-SHARGABI, 2024).
- **Raspberry Pi:** Computador de placa única (*Single-Board Computer* - SBC) com arquitetura ARM, capaz de rodar distribuições Linux completas. Em redes IoT, é frequentemente utilizado como *gateway* de borda (*Edge Gateway*) para agregar e pré-processar dados localmente antes do envio à nuvem (AL-NBHANY; ZAHARY; AL-SHARGABI, 2024).

2.2 Ameaças Cibernéticas e Vulnerabilidades na IoT

A adoção massiva de dispositivos conectados, discutida na seção anterior, expande consideravelmente a superfície de ataque disponível a agentes maliciosos. Embora a heterogeneidade de protocolos e a mobilidade dos terminais de borda tragam benefícios operacionais, essas mesmas características introduzem fragilidades que dificultam a implementação de mecanismos de defesa tradicionais. Esta seção examina, primeiramente, os desafios estruturais que comprometem a segurança na borda da rede (Seção **??**) e, em seguida, detalha a mecânica e a evolução dos ataques de Negação de Serviço Distribuída (DDoS) impulsionados por botnets, uma das ameaças mais recorrentes nesse cenário (Seção **??**).

2.2.1 Desafios Arquitetônicos de Segurança

A segurança na borda da rede enfrenta limitações físicas e operacionais que impedem a adoção de soluções de defesa de grau corporativo tradicional:

- **Recursos Computacionais Restritos:** A limitação de memória RAM e de capacidade de processamento nos microcontroladores impede o uso de criptografia assimétrica complexa ou a realização de inspeção profunda de pacotes em tempo real.

- **Credenciais Padrão e Inseguras:** Grande maioria dos dispositivos é comercializados com senhas de fábrica padronizadas (como "admin"ou "12345"), que raramente são alteradas pelos usuários finais, tornando os nós alvos fáceis para ataques de força bruta e varreduras automatizadas.

- **Obsolescência de *Firmware*:** Grande parte dos dispositivos IoT não possui mecanismos eficientes de atualização remota (*Over-The-Air* - OTA). Isso faz com que vulnerabilidades conhecidas e falhas de software permaneçam expostas por muito tempo na rede.

- **Canal de Comunicação Aberto:** A transmissão de pacotes em texto claro por redes sem fio, sem o uso de autenticação forte, facilita a interceptação e a manipulação do tráfego em ataques do tipo *Man-in-the-Middle* (MitM) (FADELE et al., 2017).

2.2.2 Ataques de Negação de Serviço Distribuída (DDoS) e *Botnets*

As vulnerabilidades comuns dos dispositivos IoT os tornam alvos fáceis para infecção por *malwares*. Quando comprometidos, esses equipamentos passam a integrar redes controladas por invasores, conhecidas como *botnets*. Essa infraestrutura distribuída é usada para executar ataques de Negação de Serviço Distribuída (DDoS), com o objetivo de esgotar a largura de banda ou a capacidade de processamento dos servidores, bloqueando o acesso de usuários legítimos (AWS, 2020).

O primeiro ataque de negação de serviço documentado ocorreu em 1996 contra o provedor Panix, utilizando a técnica de *SYN Flood*. No entanto, a proporção desses ataques cresceu drasticamente com a expansão dos dispositivos conectados. O *malware Mirai*, em 2016, foi um marco nesse cenário: ao explorar centenas de milhares de câmeras IP e roteadores com senhas padrão de fábrica, gerou um volume de tráfego de 1,2 Terabits por segundo, derrubando grandes serviços digitais no mundo todo (DONNO et al., 2018). Outro caso histórico ocorreu em 2012, quando o grupo *Anonymous* realizou ataques DDoS em protesto contra a lei SOPA, tirando do ar sites do governo dos EUA, como o do FBI e da Casa Branca (Palo Alto Networks, 2024).

De acordo com o modelo de referência *Open Systems Interconnection* (OSI), as abordagens ofensivas de DDoS classificam-se em quatro categorias primárias:

- **Ataques à Camada de Aplicação (Camada 7):** Têm como objetivo esgotar os recursos de processamento do servidor web por meio do envio de requisições aparentemente legítimas, mas custosas para o sistema. Um exemplo clássico é o *HTTP Flood*, que inunda o alvo com requisições GET ou POST (Fortinet, 2023a).

- **Ataques de Protocolo (Camadas 3 e 4):** Buscam sobrecarregar *firewalls*, balanceadores de carga e tabelas de estado do sistema operacional. O ataque *SYN Flood*, por exemplo, explora o *three-way handshake* do protocolo TCP enviando pedidos de sincronização com IPs de origem falsificados (*spoofing*). Isso força o servidor a reservar recursos para conexões que nunca são concluídas (Fortinet, 2023a).
- **Ataques Volumétricos:** Buscam esgotar a largura de banda do link de rede enviando um alto volume de tráfego. O principal exemplo é o *UDP Flood*, que direciona uma grande quantidade de datagramas para portas aleatórias do alvo. Isso sobrecarrega o servidor, que precisa alocar recursos para verificar as portas e gerar as mensagens de erro ICMP de resposta (Fortinet, 2023a).
- **Ataques Multivetoriais:** Combinam técnicas volumétricas, de protocolo e de aplicação, atuando de forma simultânea ou alternada. O objetivo é burlar regras estáticas de bloqueio e dificultar a ação dos mecanismos de defesa da rede (Fortinet, 2023a).

2.3 Detecção e Mitigação de Ameaças via Tráfego de Rede

Diante das vulnerabilidades e dos vetores de ataque apresentados na seção anterior, torna-se necessário compreender como o tráfego de rede pode ser observado e utilizado como fonte de evidências para a identificação de comportamentos anômalos. Esta seção apresenta, primeiramente, os conceitos fundamentais da Análise de Tráfego de Rede (NTA), disciplina responsável por monitorar e caracterizar os fluxos de comunicação em um ambiente de rede, e, em seguida, descreve os principais mecanismos tradicionais empregados na mitigação de ataques volumétricos e de protocolo, contextualizando as limitações que motivam a adoção de abordagens baseadas em Aprendizado de Máquina discutidas nas seções subsequentes.

2.3.1 Análise de Tráfego de Rede (NTA)

O tráfego de rede é composto pelos pacotes de dados em trânsito por um meio de comunicação, dividindo-se entre fluxos em tempo real (como *Voice over IP* (VoIP) e navegação web, críticos para a latência) e tráfego de melhor esforço ou assíncrono (como transferência de arquivos via *File Transfer Protocol* (FTP) e e-mails) (Fortinet, 2023b) (Fortinet, 2023b).

A Análise de Tráfego de Rede (NTA, do inglês *Network Traffic Analysis*), é uma disciplina técnica focada no monitoramento e inspeção em tempo real do fluxo de uma rede avaliar a qualidade do serviço, mapear gargalos operacionais e alertar administradores sobre comportamentos divergentes que indiquem violações de segurança ou início de

inundações volumétricas (Fortinet, 2023b). Historicamente impulsionada pela introdução de utilitários de captura bruta como o *tcpdump* em 1988, a NTA evoluiu de simples análises estatísticas e mineração de dados (*Data Mining*) para a incorporação moderna de algoritmos de inteligência artificial na extração de padrões complexos (**??**).

2.3.2 Mecanismos Tradicionais de Mitigação

As organizações combinam diferentes barreiras defensivas para conter ou desviar os efeitos de um ataque DDoS. Entre as contramedidas consolidadas estão:

- **Avaliação de Riscos e Rede *Anycast*:** O mapeamento de vulnerabilidades e o uso do roteamento *Anycast* permitem distribuir o tráfego de um ataque volumétrico entre vários centros de mitigação (*scrubbing centers*) espalhados geograficamente (Fortinet, 2023a).
- **Roteamento de Buraco Negro (*Blackhole Routing*):** Técnica utilizada por provedores para descartar todo o tráfego destinado a um endereço IP sob ataque. O objetivo é isolar o alvo e evitar que o volume de pacotes sobrecarregue o restante da rede (Fortinet, 2023a).
- ***Firewalls* de Aplicação Web (WAF):** Soluções que operam como *proxies* reversos para inspecionar o tráfego na Camada 7. Eles filtram requisições maliciosas ou malformadas antes que cheguem aos servidores web (Fortinet, 2023a; IBM, 2022).
- **Sistemas de Detecção e Resposta (EDR/NDR):** Ferramentas de monitoramento contínuo para *endpoints* (EDR) e redes (NDR). Elas correlacionam Indicadores de Comprometimento (IoCs) para identificar ameaças e bloquear conexões maliciosas de forma automatizada (IBM, 2022).

2.4 Inteligência Artificial e *Machine Learning* na Segurança IoT

A ineficácia das defesas baseadas em assinaturas estáticas contra ataques de dia zero (*zero-day*) ou inundações mutáveis impulsionou a adoção do Aprendizado de Máquina (ML) na automação da segurança cibernética.

2.4.1 Sistemas de Detecção de Intrusão (IDS)

Um Sistema de Detecção de Intrusão (IDS), atua como um elemento de vigilância passiva ou reativa projetado para inspecionar o tráfego de computadores ou hosts individuais, identificando violações de diretrizes de segurança e comprometimentos à integridade, confidencialidade e disponibilidade dos dados.

Os primeiros conceitos dessa tecnologia surgiram entre 1984 e 1986, quando Dorothy Denning e Peter Neumann modelaram o protótipo IDES (*Intrusion Detection Expert System*). A hipótese central do IDES definia que a conduta de um invasor no sistema diverge do comportamento de um usuário legítimo. Com base nessa premissa, os modernos Sistemas de Detecção de Intrusão em Rede (NIDS, do inglês *Network IDS*), operam analisando o histórico e as métricas capturadas em tempo real (como frequência de pacotes, tempo entre chegadas e proporção de *flags* TCP), traçando perfis de normalidade e classificando desvios como anomalias hostis (OSTEC, 2018; AZAM; ISLAM; HUDA, 2023).

2.4.2 Algoritmos Classificadores

Para a classificação supervisionada de fluxos de tráfego, a literatura como a de (MAGáN-CARRIóN DANIEL URDA; DORRONSORO, 2020) destaca um conjunto principal de algoritmos estatísticos e vetoriais:

- **Árvore de Decisão (*Decision Tree*):** Modelo estruturado em ramificações condicionais, onde cada nó faz um teste em uma *feature* da rede e divide os dados até chegar a uma classificação final (folha). Destaca-se por ser rápido e exigir pouco processamento.
- **Floresta Aleatória (*Random Forest*):** Algoritmo de *ensemble* baseado na técnica de *bagging*. Ele constrói várias árvores de decisão independentes usando subconjuntos aleatórios de dados e *features*. A classificação final é decidida por votação, o que reduz a variância e ajuda a evitar o sobreajuste (*overfitting*).
- **Máquina de Vetores de Suporte (SVM, do inglês *Support Vector Machine*):** Algoritmo que mapeia os dados em um espaço de maior dimensão usando funções de *kernel* (como o RBF). O objetivo é encontrar a melhor linha de separação (hiperplano de margem máxima) para isolar o tráfego normal dos ataques.
- ***Adaptive Boosting* (AdaBoost):** Algoritmo de *ensemble* que treina uma sequência de classificadores simples (geralmente árvores de decisão de um único nível, chamadas de *stumps*). A cada rodada, o modelo aumenta o peso das amostras classificadas incorretamente, focando nos erros anteriores para melhorar o resultado final.
- **Perceptron de Múltiplas Camadas (MLP, do inglês *Multilayer Perceptron*):** Modelo tradicional de Rede Neural Artificial formado por camadas de neurônios interligados. O algoritmo ajusta seus parâmetros usando o método de *backpropagation* (retropropagação de erro) para identificar padrões complexos no tráfego da rede.

2.5 Otimização de Modelos para Ambientes Restritos (*Edge Computing*)

O avanço na precisão dos modelos de Machine Learning e Deep Learning é frequentemente acompanhado por um aumento exponencial na sua complexidade matemática e na demanda por alocação de memória. Em ambientes de computação em nuvem ou servidores dedicados de segurança, esse custo computacional é absorvido com facilidade. No entanto, quando a aplicação do IDS se desloca para a borda da rede , rodando em *gateways* locais, roteadores domésticos ou microcontroladores industriais como o Raspberry Pi, surge um conflito direto entre a precisão preditiva e a viabilidade do hardware.

Sistemas modernos de captura de tráfego de rede produzem bases de dados com alta dimensionalidade, gerando dezenas de métricas estatísticas por fluxo. Ingerir, armazenar em buffer e computar matrizes com 40, 50 ou mais *features* simultaneamente por pacote na borda resulta em dois problemas severos:

1. **A Maldição da Dimensionalidade (*Curse of Dimensionality*):** O excesso de *features* irrelevantes ou colineares insere ruído estatístico no processamento, degradando a capacidade dos algoritmos paramétricos (como Redes Neurais e SVM) e aumentando a taxa de alarmes falsos no tráfego legítimo.
2. **Alto Consumo de Recursos e Latência:** Processar muitas *features* em tempo real sobrecarrega a CPU e a pouca memória RAM dos dispositivos IoT. Isso atrasa a análise dos pacotes e pode causar travamentos ou prejudicar a Qualidade de Serviço (QoS) da rede.

Para que a segurança cibernética seja efetiva na borda, os modelos de IA precisam passar por processos rigorosos de otimização arquitetônica e engenharia de *features*. Torna-se indispensável identificar e isolar apenas o subconjunto mínimo de características (*features*) da rede que contenha o sinal puro do ataque, descartando variáveis redundantes. A otimização reduz a carga de operações de ponto flutuante, viabilizando a execução de inferências instantâneas com consumo de memória quase nulo em microcontroladores embarcados.

2.6 Interpretabilidade e Seleção de *features*: O Método SHAP

A maioria dos algoritmos com alta capacidade preditiva na cibersegurança, operam sob a natureza de "caixas-pretas"(*black-boxes*). A sua lógica interna de centenas de árvores simultâneas, não é legível diretamente por analistas humanos. No contexto da cibersegurança, a falta de transparência é inaceitável: administradores de redes precisam

compreender por que um fluxo foi rotulado como intrusão, auditar as causas dos alertas e garantir que o modelo não tome decisões baseadas em ruídos de captura.

Para resolver o problema da falta de transparência e, ao mesmo tempo, viabilizar a otimização computacional na borda, utiliza-se o método SHAP, proposto por Lundberg e Lee (LUNDBERG; LEE, 2017). O SHAP baseia-se nos Valores de Shapley, um conceito da Teoria dos Jogos Cooperativos usado para calcular a contribuição matemática exata de cada "jogador"(neste caso, cada *feature* de rede) na decisão final do classificador.

Diferentemente de métricas heurísticas internas, como a importância baseada na impureza de Gini nativa das árvores de decisão, o SHAP é padrão para a arquitetura do modelo. Ele possibilita auditar e comparar o impacto real desses *features* estatísticas em classificadores de naturezas distintas sob o mesmo critério matemático. No cálculo prático dos valores de Shapley em redes multivariadas, implementam-se duas estratégias de processamento (LUNDBERG; LEE, 2017):

- ***TreeExplainer*:** Algoritmo otimizado especificamente para modelos baseados em árvores (*Decision Tree* e *Random Forest*), que varre as fronteiras de decisão da topologia das árvores para calcular os valores SHAP em tempo polinomial rápido.
- ***KernelExplainer*:** Estratégia aproximada e modelo padrão aplicada a classificadores vetoriais e paramétricos (SVM e Redes Neurais), operando sobre subconjuntos de fundo de amostras para contornar gargalos combinatórios na inferência de importância.

A aplicação do método SHAP no projeto de um IDS para a Internet das Coisas possui um duplo objetivo. Além de prover a explicabilidade necessária para justificar bloqueios de tráfego, o ranqueamento global das variáveis de maior contribuição marginal atua como uma ferramenta avançada de Seleção de *features* (*Feature Selection*). Ao reter apenas as variáveis no topo do ranking SHAP (como as Top 10 ou Top 15 características mais relevantes) e suprimir o restante da matriz de tráfego, enxuga-se drasticamente o espaço vetorial, reduzindo o custo computacional dos algoritmos e viabilizando o embarque de um sistema IDS interpretável, veloz e altamente preciso nos dispositivos de borda da IoT.

3 Trabalhos Correlatos

Diversos trabalhos na literatura buscam desenvolver Sistemas de Detecção de Intrusão (IDS) para a Internet das Coisas (IoT), com o desafio de manter uma alta precisão sem sobrecarregar dispositivos com poucos recursos computacionais.

Este capítulo analisa cinco estudos relacionados à explicabilidade de modelos, aprendizado em conjunto (*Ensemble Learning*) e compressão de tráfego na borda. Para cada trabalho, apresentamos a proposta, a metodologia e os resultados. Ao apontar as limitações dessas soluções, principalmente o alto custo computacional e a falta de transparência nas decisões.

3.1 Trabalho 1 – *Toward Enhanced Attack Detection and Explanation in Intrusion Detection System-Based IoT Environment Data*

.

Proposta: O estudo de (LE et al., 2023), publicado no periódico *IEEE Access*, abordou o desafio da falta de transparência preditiva em Sistemas de Detecção de Intrusão para a Internet das Coisas, propondo uma arquitetura focada na união entre alta precisão de classificação e Inteligência Artificial Explicável (XAI, do inglês *Explainable AI*) (LE et al., 2023). Os autores argumentam que os métodos tradicionais de aprendizado de máquina operam como "caixas-pretas" (*black-boxes*), impedindo que analistas compreendam os fatores que levam o sistema a classificar um fluxo como benigno ou malicioso.

Método: Para resolver essa limitação, a pesquisa propôs um modelo de *Ensemble Blending* (uma arquitetura de dois níveis que combina *Gradient Boosting*, *Random Forest* e Árvore de Decisão), integrado às técnicas de explicabilidade LIME (*Local Interpretable Model-Agnostic Explanations*) e explicações contrafactuais (LE et al., 2023). Os testes foram realizados nos *datasets* **CICIoT2023** e IoTID20. No pré-processamento do CICIoT2023, os autores aplicaram o critério de diminuição média de impureza (MDI, do inglês *Mean Decrease in Impurity*) para selecionar as seis *features* mais importantes (`IAT`, `Magnitude`, `Total Size`, `Min`, `Flow Duration` e `Tot sum`).

Resultados e Limitações: Na classificação multiclasse, o modelo alcançou uma acurácia de 99,51% e AUC-ROC de 1,0000 na base CICIoT2023, superando os classificadores individuais (LE et al., 2023). O método LIME conseguiu identificar com sucesso quais variáveis influenciaram as decisões do modelo. No entanto, a abordagem apresenta

problemas para ser aplicada na borda da rede: a exigência de processamento e memória para treinar e rodar um modelo *Blending* de vários níveis é muito alta. O estudo registrou um tempo de inferência de 3,89 segundos, uma latência inadequada para bloquear ataques DDoS em tempo real na IoT.

3.2 Trabalho 2 – *Ensemble-Learning Framework for Intrusion Detection to Enhance Internet of Things' Devices Security* (Alotaibi & Ilyas, 2023)

Proposta: O estudo de (ALOTAIBI; ILYAS, 2023), publicado no periódico *Sensors*, propôs um *framework* de aprendizado de máquina em conjunto (*Ensemble Learning*) desenhado especificamente para aprimorar a segurança de dispositivos IoT contra ataques de rede, com foco em inundações volumétricas e exaustão de protocolo (ALOTAIBI; ILYAS, 2023). O objetivo central dos autores foi construir um modelo de alta generalização capaz de superar a instabilidade e a variância de classificadores singulares quando submetidos a tráfegos heterogêneos.

Método: A metodologia utilizou uma estratégia de votação majoritária (*Majority Voting*), combinando os algoritmos *Random Forest*, *XGBoost* e *LightGBM* (ALOTAIBI; ILYAS, 2023). Os testes foram realizados nos *datasets* públicos IoTID20 e BoT-IoT. Na etapa de engenharia de *features*, os autores aplicaram técnicas estatísticas tradicionais, como o índice de impureza de Gini e o teste Qui-Quadrado (χ^2), selecionando um conjunto de 25 a 30 variáveis para treinar os modelos.

Resultados e Limitações: O modelo de *ensemble* alcançou uma acurácia de 99,45% e uma taxa de detecção acima de 99,2% para ataques DDoS volumétricos na base BoT-IoT (ALOTAIBI; ILYAS, 2023). No entanto, o estudo apresenta duas restrições importantes para o uso na borda da rede. A primeira é que o classificador funciona como uma "caixa-preta", sem mecanismos de interpretabilidade (XAI), o que impede a auditoria dos alertas gerados. A segunda é que o uso de 25 a 30 *features* ao mesmo tempo, somado à execução paralela de três algoritmos pesados, gera uma sobrecarga computacional que inviabiliza a aplicação do sistema em microcontroladores de baixo custo.

3.3 Trabalho 3 -*Bridging the Interpretability Gap: A SHAP - Enhanced Framework for Intrusion Detection in Cybersecurity* (Goyal et al., 2025)

Proposta: O estudo conduzido por (GOYAL; THAKUR; QADEER, 2025), publicado no *International Journal of Management and Data Intelligence*, investigou a necessidade urgente de desenvolver Sistemas de Detecção de Intrusão baseados em Inteligência Artificial Explicável (GOYAL; THAKUR; QADEER, 2025). Os autores enfatizam que a falta de transparência inerente aos modelos complexos impede a sua implantação em ambientes cibernéticos críticos, onde a confiabilidade da decisão computacional é indispensável.

Método: Para resolver a falta de transparência, a pesquisa propôs um *framework* de detecção utilizando o algoritmo *SHapley Additive exPlanations* (SHAP) (GOYAL; THAKUR; QADEER, 2025). O estudo avaliou o desempenho de classificadores supervisionados, comparando algoritmos baseados em árvore (Árvore de Decisão, *Random Forest* e *XGBoost*) com Redes Neurais Convolucionais (CNN). O *dataset* utilizado para os testes foi o NSL-KDD (uma versão melhorada do clássico KDD Cup 99). O SHAP foi empregado para gerar explicações globais e locais após a classificação (*post-hoc*), calculando o impacto de cada *feature* na separação entre o tráfego normal e os ataques. Com isso, os autores identificaram variáveis críticas como `Destination Port`, `Bwd Packet Length Min` e `Total Length of Fwd Packets`.

Resultados e Limitações: A integração da explicabilidade visual e da seleção de *features* via SHAP proporcionou um incremento médio de 3,4% na acurácia de detecção em comparação aos critérios tradicionais de entropia (GOYAL; THAKUR; QADEER, 2025). Contudo, em sua própria discussão, os autores reconhecem que a dependência de bases de dados estáticas e historicamente antigas (como NSL-KDD) compromete severamente a capacidade do *framework* em generalizar e gerenciar a evolução dinâmica das ameaças contemporâneas, incentivando a comunidade a validar essas técnicas em redes IoT modernas. Ademais, o estudo restringe o SHAP a uma ferramenta de auditoria visual, sem explorar os limites de corte dimensional para máxima economia na borda.

3.4 Trabalho 4 – *Lightweight IoT Intrusion Detection via Feature and Sample Reduction With Multi-Client Ensemble for Zero-Day Attacks*

.

Proposta: Um estudo recente que busca equilibrar a restrição de hardware e a eficiência computacional é o de Zhang et al. (2026), que compartilha a mesma base empírica adotada nesta monografia: o conjunto de dados moderno **CICIoT2023** (ZHANG et al., 2026). Os autores propuseram um modelo distribuído de detecção de intrusões desenhado para operar com latência e consumo energético mínimos diretamente na borda da rede (*Edge Computing*).

Método: Para suprimir a alta dimensionalidade e mitigar a assimetria da base de dados, implementaram um *pipeline* de compressão multiestágio integrado por três fases sequenciais (ZHANG et al., 2026):

1. **Seleção de *features* por Informação Mútua (MI):** Descartaram as 19 colunas menos informativas do CICIoT2023, retendo as 20 variáveis de maior correlação com os ataques.
2. **Redução por PCA:** As 20 características foram transformadas por Análise de Componentes Principais (PCA), projetando ortogonalmente o espaço em apenas 9 componentes principais ($PC_1, PC_2, \ldots, PC_9$) que preservam 90% da variância.
3. **Compressão via Clusterização:** O espaço reduzido foi submetido ao agrupamento *k-means* ($k = 13$), descartando registros redundantes e mantendo apenas os centróides mais representativos (retendo de 90% a 10% das amostras).

Na modelagem preditiva, adotaram o classificador *LightGBM* executado localmente em clientes de borda, combinando o conhecimento via meta-aprendizado de conjunto (*Hard Voting*, F1-Score e *Stacking*).

Resultados e Limitações: Os testes em hardware embarcado real (*Raspberry Pi 4 Model B*) exibiram resultados expressivos: ao reter apenas 10% das amostras, o consumo total de energia caiu de 0,002015 kWh (treino direto) para apenas 0,000145 kWh (redução de 92,8%), mantendo taxa de revocação (*Recall*) superior a 97% e processando até 173.005 pacotes por segundo (ZHANG et al., 2026). Contudo, o método sofre com uma penalidade severa provocada pelo PCA: a destruição da semântica física original do tráfego. Como a decisão baseia-se em componentes abstratos (e.g., $PC_1 > 0,85$), o sistema perde totalmente a interpretabilidade causal do ataque.

3.5 Trabalho 5 – *A Novel Intrusion Detection Approach Using Machine Learning Ensemble for IoT Environments*

.

Proposta: O estudo de (VERMA et al., 2021), publicado no periódico *Applied Sciences*, propôs uma abordagem de detecção de intrusão orientada à proteção de ambientes IoT contra ataques cibernéticos de dia zero (*zero-day*) e inundações de rede. O objetivo central foi estruturar um classificador binário robusto capaz de filtrar e descartar o tráfego malicioso na borda, abordando simultaneamente o desbalanceamento de classes e a alta dimensionalidade das transmissões de dados.

Método: A metodologia utilizou técnicas de aprendizado em conjunto (*Ensemble Learning*), combinando os algoritmos *Random Forest* e *Gradient Boosting Machine* (GBM), implementados com a biblioteca *CatBoost* no ambiente Python. Para otimizar os hiperparâmetros, empregaram uma busca aleatória (*Randomized Search CV*). O modelo foi avaliado sobre o conjunto de dados CSE-CIC-IDS2018-V2, onde os fluxos brutos de 15 tipos de tráfego foram agrupados em 9 categorias principais e remapeados para uma classificação binária (ataque versus normal). Na engenharia de *features*, os autores implementaram uma subamostragem aleatória (*undersampling*) no tráfego benigno para equilibrar as classes e utilizaram o algoritmo *XGBoost* aliado ao critério de impureza de Gini para calcular o escore de importância e extrair as características predominantes.

Resultados e Limitações: A abordagem de *ensemble* proposta obteve um desempenho expressivo na base CSE-CIC-IDS2018-V2, alcançando uma acurácia global de 98,27%, precisão de 96,40% e taxa de revocação (*Recall*) de 95,70%. A avaliação de latência computacional em um dispositivo de borda (*Raspberry Pi 3B+*) demonstrou um tempo de resposta de $9,22 \times 10^{-7}$ segundos por instância. Contudo, em termos de comparação com esta monografia, o trabalho apresenta duas ressalvas metodológicas: primeiro, o estudo limitou-se à classificação binária (normal vs. ataque), não resolvendo o discernimento de classes dos vetores DDoS (como a diferenciação entre *HTTP Flood*, *SYN Flood* e *UDP Flood*); segundo, a seleção de *features* foi fundamentada puramente em escores heurísticos internos (impureza de Gini), falhando em fornecer interpretabilidade física causal (*XAI*) sobre o tráfego ou em testar subconjuntos dimensionais mínimos (como Top 5 e Top 10 características) para economizar processamento.

3.6 Análise da Literatura e Posicionamento Teórico (Justificativa)

A revisão da literatura mostra que as pesquisas em cibersegurança para IoT alcançaram avanços importantes na redução do custo computacional na borda (ZHANG et al., 2026). Contudo, ainda há uma lacuna metodológica que dificulta a adoção dessas tecnologias em infraestruturas reais corporativas e industriais.

Embora (ZHANG et al., 2026) alcancem uma eficiência energética expressiva no *Raspberry Pi*, a aplicação do algoritmo PCA resulta na perda do significado original dos pacotes. Com isso, quando um *gateway* IoT gera um alerta de ataque DDoS, a decisão é

baseada em um vetor abstrato (como $PC_1 > 0,85$ e $PC_3 < -0,12$) (ZHANG et al., 2026). Isso impede que os administradores saibam qual parâmetro real da rede disparou o alarme — como um pico na taxa de transmissão `Rate`, uma anomalia no tempo de resposta `IAT` ou o excesso da flag `syn_flag_number`.

Por outro lado, (ALOTAIBI; ILYAS, 2023) utilizam 30 variáveis em *ensembles* pesados sem oferecer interpretabilidade, limitando a otimização ao índice de Gini para classificação binária, sem testar o desempenho com conjuntos menores de *features* (VERMA et al., 2021). Já (LE et al., 2023) e (GOYAL; THAKUR; QADEER, 2025) usam as técnicas XAI (LIME e SHAP) apenas como ferramentas visuais após a classificação (*post-hoc*), sem aproveitar seu potencial para reduzir o tamanho do modelo (LE et al., 2023; GOYAL; THAKUR; QADEER, 2025). Em ambientes críticos, como a Internet das Coisas Médicas (IoMT) ou automação industrial, usar sistemas "caixa-preta" sem rastreabilidade é um risco operacional, pois dificulta auditorias e a criação de regras específicas em *firewalls*.

É para preencher essa lacuna técnica que este Trabalho de Conclusão de Curso propõe sua abordagem. Para atuar diretamente na borda com a mesma eficiência defendida por (ZHANG et al., 2026), este trabalho substitui o uso do PCA pelo SHAP, aplicando-o ativamente na Seleção de *features (Feature Selection)*.

Baseado na Teoria dos Jogos Cooperativos, o SHAP calcula matematicamente a contribuição de cada variável na decisão do modelo. Em vez de transformar os dados de tráfego em componentes abstratos e difíceis de interpretar, esta pesquisa reduz a dimensionalidade mantendo apenas as Top 10 e Top 15 características originais da rede que mais influenciam a detecção dos ataques.

Dessa forma, a metodologia proposta supera essas limitações ao focar em quatro pilares:

- **Leveza Computacional na Borda:** O uso de no máximo 15 *features* originais evita o problema de alta dimensionalidade, otimizando o consumo de memória RAM e de processamento sem precisar de componentes abstratos.
- **Desempenho Preditivo e Estabilidade:** Mantendo os dados originais, modelos de conjunto como a Floresta Aleatória (*Random Forest*) alcançam o teto de acurácia da base CICIoT2023 (99,80%). Além disso, a redução de ruído estabiliza a convergência de Redes Neurais (MLP).
- **Interpretabilidade e Auditabilidade:** Todo alerta gerado na borda é compreensível. O analista consegue visualizar em tempo real quais métricas da rede motivaram a classificação e o bloqueio do ataque.
- **Superação do *Baseline*:** O estudo comprova na prática que a abordagem proposta é superior, especialmente ao demonstrar a queda drástica de acurácia (30,81%) de

modelos não supervisionados ao lidar com tráfegos densos, justificando a adoção de florestas supervisionadas otimizadas por SHAP.

3.7 Quadro Comparativo e Síntese Bibliográfica

Para consolidar o posicionamento desta monografia em relação às quatro pesquisas primárias analisadas, a Tabela 1 apresenta uma síntese comparativa considerando o ano de publicação, o conjunto de dados utilizado, a abordagem algorítmica, o método de redução dimensional, o nível de interpretabilidade oferecido e a viabilidade prática para embarque em hardware restrito de borda.

Tabela 1 – Comparativo Analítico entre os Trabalhos Correlatos

Estudo	**Ano**	**Dataset**	**Algoritmos / Abordagem**	**Viabilidade no Hardware**
Le et al.	2023	CICIoT2023	*Blending* + LIME/Contra.	Baixa (Modelo pesado)
Alotaibi & Ilyas	2023	BoT-IoT	RF, XGBoost, *LightGBM*	Baixa (Alta sobrecarga)
Goyal et al.	2025	NSL-KDD	RF, XGBoost, CNN + SHAP	Moderada (Base antiga)
Zhang et al.	2026	CICIoT2023	*LightGBM* + *Ensembles*	Alta (Baixo consumo)
Verma et al.	2021	CSE-CIC-IDS2018	RF + GBM (*CatBoost*)	Alta (Binária / $0,92\,\mu$s)[cite: 4]
Este Trabalho	2026	CICIoT2023	5 classificadores + Consenso SHAP	Não testada (redução dimensional)

4 Método

Este capítulo descreve o método adotado para avaliar os modelos de aprendizado de máquina e de classificação aplicados à detecção fluxos de ataque DDoS. São apresentados o *dataset* utilizado, as etapas de pré-processamento, os algoritmos aplicados com suas respectivas configurações, análise SHAP, e as métricas utilizadas na avaliação dos resultados.

4.1 Dataset

Os dados trabalhados são do conjunto de dados público CICIoT2023, elaborado pelo *Canadian Institute for Cyber- security* (CIC), que fornece capturas rotuladas de tráfego proveniente de múltiplos dispositivos inteligentes sob diversas tipologias de ataques. Apresenta 47 colunas com dados de 33 tipos de ataques diferentes divididos em 7 categorias (*DDoS, DoS, Reconnaissance, Web-based, Brute Force, Spoofing e Mirai*)

O espaço amostral do experimento foi construído por 25.000 instâncias, distribuídas cinco blocos de 5.000 registros da categoria DDoS mas de tipologias diferentes:

- **Tráfego Benigno (*Benign*):** Representa o comportamento padrão de comunicação da rede IoT, englobando requisições periódicas de telemetria, sensoriamento ambiental e resposta de atuadores sob protocolos adaptados.
- ***DDoS HTTP Flood:*** Vetor de ataque focado na exaustão de recursos da camada de aplicação (Camada 7 do Modelo OSI), no qual o atacante sobrecarrega servidores web encaminhando um volume massivo de requisições maliciosas aparentemente legítimas.
- ***DDoS SYN Flood:*** Agressão direcionada à camada de transporte, explorando a mecânica do aperto de mão de três vias (*three-way handshake*) do protocolo TCP mediante o envio contínuo de pedidos de sincronização com endereços falsificados, visando esgotar as tabelas de estado dos roteadores.
- ***DDoS UDP Flood:*** Ataque volumétrico de força bruta que inunda portas aleatórias do dispositivo alvo com um enxame de datagramas sem conexão, consumindo integralmente a largura de banda disponível do canal de comunicação.
- ***Mirai-udpplain:*** Tráfego de intrusão gerado pela *botnet* Mirai, ilustrando a ação de *malwares* especializados em infectar dispositivos IoT vulneráveis para orquestrar ataques ofensivos de alto tráfego em larga escala.

4.2 Pipeline de Procesamento

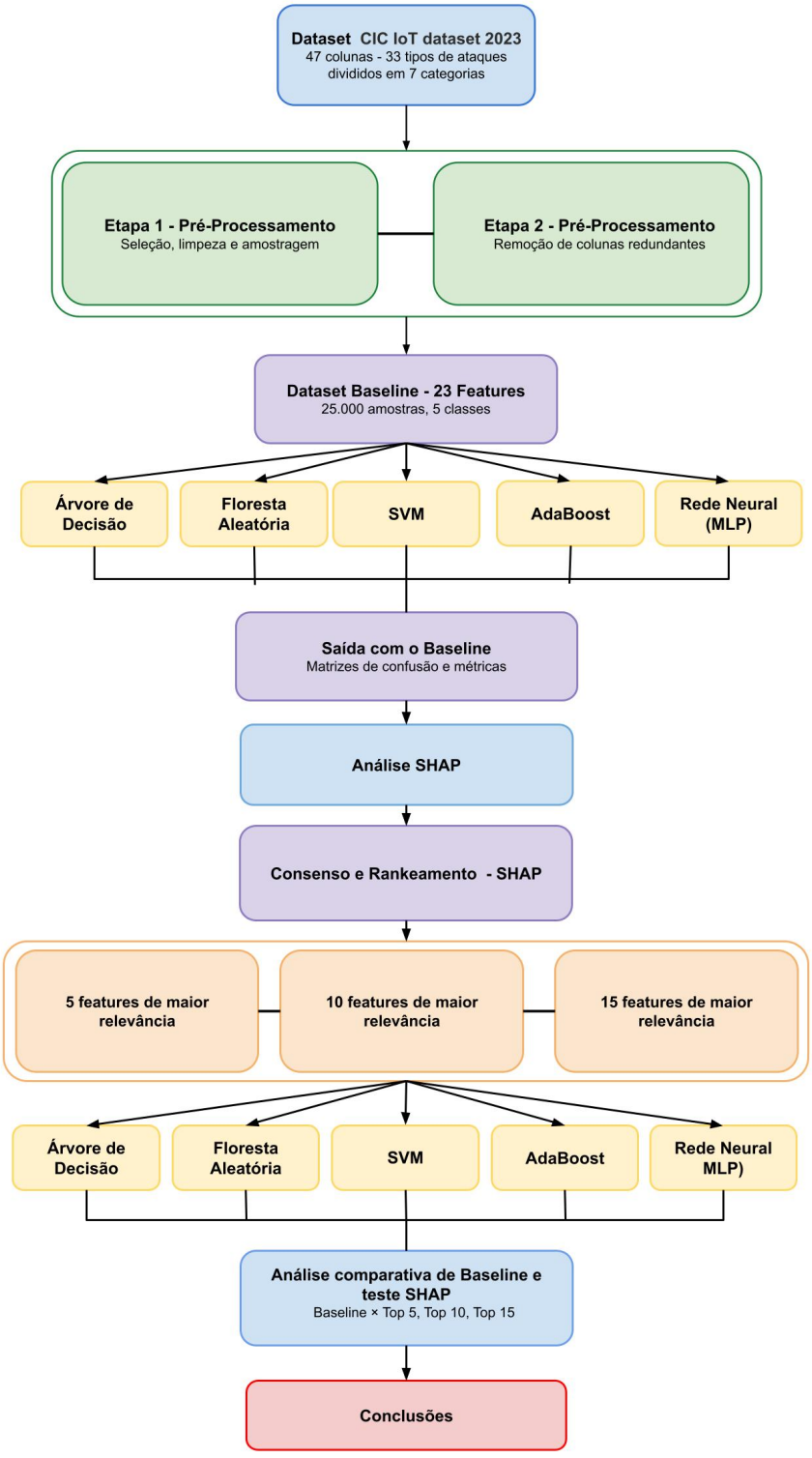

Figura 1 – Pipeline completo

Fonte: elaborado pelo autor

4.2.1 Etapa 1 - Pré-Processamento

De posse dos arquivos segmentados, aplicou-se uma rotina de higienização na qual 47 *features* foram suprimidas, sendo as colunas *"Telnet", "SMTP", "Protocol_Type", "Min", "Number", "Tot_size", "LLC", "ARP"*, as quais têm médias de valores extremamente baixas ou quase zero, informações redundantes ou repetidas, presentes em colunas que se mantiveram no trabalho.

Uma etapa corretiva consistiu no tratamento de formatações numéricas inconsistentes. Variáveis críticas como *Rate, Std, Variance e IAT* apresentavam caracteres alfanuméricos entre os dígitos, incluindo pontos de separação de milhar e vírgulas decimais. Para regularizar essas distribuições, aplicaram-se funções de expressão regular (regex) encarregadas de remover os separadores de milhar e padronizar a separação decimal em ponto, realizando, em seguida, o casting de todas as características para o tipo numérico de ponto flutuante de precisão simples (float32). Concluída a tipagem, foi executada uma varredura para localizar e descartar imediatamente qualquer linha que apresentasse valores ausentes (NaN) ou quantidades numéricas infinitas (Inf e -Inf) resultando em um *dataframe* concatenado e quase completamente limpo.

4.2.2 Etapa 2 - Pré-Processamento

Carregado o *dataset* da etapa anterior, foi identificada e separada a coluna de rótulo dos demais dados. Em seguida, procedeu-se à remoção de colunas constantes e altamente correlacionadas. Correlação de Pearson: calculou-se a matriz de correlação absoluta entre os *features* remanescentes, adotando-se um limiar de corte conservador fixado em 80% (`LIMITE_CORRELACAO = 0.80`). O processo resultou em dois arquivos de saída: *Features_Limpas.csv*, contendo somente os registros sem a coluna *Label* de classificação, e *Classes_Limpas.csv*, contendo somente as classes. As *features* restantes totalizaram 23 (*Header_Length, Time_To_Live, Rate, fin_flag_number, syn_flag_number, rst_flag_number, psh_flag_number, ece_flag_number, ack_count, HTTPS, DNS, SSH, IRC, DHCP, ICMP, IGMP, IPv, Tot_sum, Max, AVG, Std, IAT, Variance*), consolidando o *Baseline*.

4.3 Especificações e Treinamento dos Classificadores

Superada a etapa de pré-processamento, a pesquisa avaliou comparativamente cinco modelos supervisionados de aprendizado de máquina, representativos dos principais metodos de classificação, implementados na biblioteca Scikit-Learn:

Tabela 2 – Modelos de classificação utilizados e suas configurações experimentais

Modelo	**Descrição**	**Configuração (Hiperparâmetros)**
Árvore de Decisão	Modelo de regras simbólicas hierárquicas, fundamentado na divisão ortogonal de espaço por limiares de entropia.	`random_state=42` `class_weight='balanced'` `criterion='gini'` (padrão)
Floresta Aleatória	Algoritmo de ensemble baseado em bagging, que agrega centenas de árvores independentes treinadas sobre subconjuntos aleatórios para mitigar a variância do preditor.	`n_estimators=100` `random_state=42` `class_weight='balanced'` `n_jobs=-1`
SVM	Classificador discriminativo configurado com função de kernel de Base Radial (RBF), voltado para traçar hiperplanos de margem máxima em dimensões superiores.	`kernel='rbf'` (padrão) `probability=True` `random_state=42` `class_weight='balanced'`
AdaBoost	Meta-algoritmo de ensemble boosting, estruturado pela sequência iterativa de classificadores fracos que penalizam e reajustam progressivamente os pesos das instâncias classificadas incorretamente.	`n_estimators=100` `random_state=42`
MLP	Arquitetura de Rede Neural Artificial de aprendizado profundo, projetada com três camadas ocultas interconectadas de 128, 64 e 32 neurônios, operando com função de ativação não linear ReLU e otimizador matricial Adam.	`hidden_layer_sizes=(128,64,32)` `activation='relu'` `solver='adam'` `max_iter=100` `early_stopping=True` `random_state=42`

Fonte: elaborado pelo autor.

Para complementar a Tabela 2, descreve-se a seguir o papel de cada hiperparâmetro configurado nos cinco classificadores.

Parâmetros comuns a múltiplos modelos:

- **`random_state=42`:** fixa a semente do gerador de números pseudoaleatórios utilizado internamente por cada algoritmo (na divisão dos dados, na inicialização de pesos ou na amostragem de subconjuntos), garantindo que o experimento seja reprodutível – execuções repetidas do mesmo código produzem sempre o mesmo resultado.
- **`class_weight='balanced'`:** ajusta automaticamente o peso atribuído a cada classe durante o treinamento, de forma inversamente proporcional à sua frequência no conjunto de dados. Isso penaliza mais os erros cometidos nas classes minoritárias, redu-

zindo o viés do classificador em favor das classes majoritárias. Parâmetro presente na Árvore de Decisão, na Floresta Aleatória e na SVM.

Parâmetros específicos de cada classificador:

- **`criterion='gini'`** (Árvore de Decisão): critério utilizado para avaliar a qualidade de cada divisão de nó, medindo o grau de impureza das classes resultantes. Corresponde ao valor padrão do *scikit-learn* – o modelo não foi configurado para utilizar o critério de entropia como alternativa.
- **`n_estimators=100`** (Floresta Aleatória e AdaBoost): número de unidades de aprendizado combinadas no *ensemble* – 100 árvores treinadas de forma independente na Floresta Aleatória, ou 100 classificadores fracos ajustados sequencialmente no AdaBoost.
- **`n_jobs=-1`** (Floresta Aleatória): paraleliza o treinamento das árvores utilizando todos os núcleos de processamento disponíveis, reduzindo o tempo total de treinamento sem alterar o resultado obtido.
- **`kernel='rbf'`** (SVM): define a função de Base Radial (*Radial Basis Function*) como núcleo de transformação, mapeando implicitamente os dados de entrada para um espaço de maior dimensão, no qual classes não linearmente separáveis no espaço original podem ser separadas por um hiperplano. Corresponde ao valor padrão do *scikit-learn*.
- **`probability=True`** (SVM): habilita a calibração interna do modelo, realizada por validação cruzada, necessária para que a SVM produza estimativas de probabilidade por classe – e não apenas rótulos discretos –, exigidas pelo cálculo da métrica AUC-ROC.
- **`hidden_layer_sizes=(128,64,32)`** (MLP): define a arquitetura da rede neural, com três camadas ocultas contendo, respectivamente, 128, 64 e 32 neurônios, em ordem decrescente de largura.
- **`activation='relu'`** (MLP): função de ativação não linear (*Rectified Linear Unit*) aplicada às camadas ocultas, definida como $f(x) = \max(0, x)$.
- **`solver='adam'`** (MLP): algoritmo de otimização utilizado para ajustar os pesos da rede durante o treinamento, com taxa de aprendizado adaptativa calculada individualmente para cada parâmetro.
- **`max_iter=100`** (MLP): número máximo de épocas (passagens completas pelo conjunto de treino) permitidas durante o ajuste dos pesos da rede.

- `early_stopping=True` (MLP): interrompe o treinamento antes de atingir `max_iter` caso o desempenho medido em um subconjunto de validação interno deixe de melhorar, prevenindo o sobreajuste (*overfitting*) do modelo.

Não foi realizada busca de hiperparâmetros (*grid search* ou similar) para nenhum dos cinco classificadores. Os parâmetros estruturais (`criterion`, `kernel`, `activation`, `solver`) seguem o padrão do *scikit-learn*; os demais decorrem de necessidades metodológicas pontuais, como `class_weight='balanced'` (tratamento de classes) e `probability = True` (cálculo do AUC-ROC). Essa configuração foi fixada uma única vez e reaplicada de forma idêntica nos quatro cenários experimentais, isolando o efeito da variação do espaço de atributos sobre o desempenho de cada modelo.

4.4 Explicabilidade via SHAP

Para reduzir a falta de transparência típica de modelos frequentemente chamados de "caixas-pretas", o pipeline incorporou a técnica de interpretabilidade agnóstica *SHapley Additive exPlanations* (SHAP), proposta por Lundberg e Lee (2017). O método se baseia na teoria dos jogos e calcula, para cada *feature* de rede, o quanto ele contribuiu para a predição do modelo. Essa contribuição é chamada de valor de Shapley e é calculada considerando todas as combinações possíveis entre os *features*. Isso garante uma propriedade importante: somando os valores de Shapley de todos os *features* a um valor-base (a predição média do modelo), chega-se exatamente ao valor previsto para aquela amostra.

O SHAP permite duas formas de análise. No nível individual (interpretação local), é possível entender por que uma amostra específica de tráfego foi classificada como benigna ou como um tipo de ataque. No nível do conjunto de dados inteiro (interpretação global), os valores de Shapley de todas as amostras são agregados para identificar quais *features* mais influenciam as decisões dos classificadores. É importante destacar que essa influência é estatística, e não causal: o SHAP mostra o que o modelo aprendeu a valorizar em suas decisões, não necessariamente o que de fato causa o ataque na rede.

4.5 Interpretabilidade SHAP

- **Gráficos de Resumo Individual (*SHAP Summary Plots*):** Para cada algoritmo testado, o sistema computou os valores de Shapley e projetou um gráfico de dispersão horizontal das características mais impactantes. A posição do ponto no eixo horizontal indica se aquela variável empurrou a decisão do modelo a favor ou contra a classificação de um ataque cibernético, permitindo compreender a lógica interna de aprendizados complexos.

- **Gráfico Consolidado de Ocorrências Globais:** Para eliminar o viés algorítmico de um único classificador, desenvolveu-se uma rotina de agregação que processa os relatórios SHAP individuais de todos os cinco modelos. O algoritmo compila um ranking global das 15 características mais frequentes e de maior impacto numérico entre todas as arquiteturas, projetando um gráfico de barras horizontais (*Top 15 Features*). Essa saída apresenta visualmente a contagem de recorrência de cada *feature* em cada decisão dos modelos, acompanhada do valor médio global de contribuição SHAP, fornecendo a comprovação matemática empírica para a seleção das variáveis utilizadas nos cenários de dimensionalidade reduzida.

4.6 Métricas de Avaliação de Desempenho dos Classificadores

A verificação empírica da eficácia dos cinco classificadores de *Machine Learning* baseou-se na construção de Matrizes de Confusão para a visualização dos acertos e erros de classificação, complementadas pelo cálculo computacional de cinco métricas estatísticas de desempenho multiclasse. Nas fórmulas a seguir, k representa o número de classes, e VP_i, FP_i, FN_i e VN_i denotam, respectivamente, os verdadeiros positivos, falsos positivos, falsos negativos e verdadeiros negativos referentes à classe i.

1. **Acurácia Global (Accuracy):** Proporção geral do total de fluxos de rede capturados que foram classificados corretamente em suas respectivas categorias.

$$\text{Acurácia} = \frac{\sum_{i=1}^{k} VP_i}{\text{Total de Amostras}} \tag{4.1}$$

2. **Precisão Macro (*Precision*):** Capacidade do classificador de não atribuir o rótulo de uma agressão cibernética a um fluxo de conexão normal, calculada pela média aritmética não ponderada da precisão de todas as cinco classes.

$$\text{Precisão}_i = \frac{VP_i}{VP_i + FP_i} \qquad \Rightarrow \qquad \text{Precisão}_{\text{Macro}} = \frac{1}{k}\sum_{i=1}^{k} \text{Precisão}_i \tag{4.2}$$

3. **Revocação Macro (Recall / Sensibilidade):** Proporção de fluxos de ataques reais ou comunicações benignas que foram identificados com precisão pelo sistema de segurança na borda.

$$\text{Revocação}_i = \frac{VP_i}{VP_i + FN_i} \qquad \Rightarrow \qquad \text{Revocação}_{\text{Macro}} = \frac{1}{k}\sum_{i=1}^{k} \text{Revocação}_i \tag{4.3}$$

4. **Escore F1 Macro (*F1-Score*):** Média harmônica equilibrada entre a Precisão e a Revocação, indicando a estabilidade do modelo e penalizando severamente desempenhos que gerem altos índices de falsos positivos ou falsos negativos.

$$F1_{\text{Macro}} = \frac{1}{k}\sum_{i=1}^{k} 2 \cdot \frac{\text{Precisão}_i \cdot \text{Revocação}_i}{\text{Precisão}_i + \text{Revocação}_i} \tag{4.4}$$

5. **Área Sob a Curva ROC (*AUC-ROC Multiclass*):** Medida probabilística calculada sob a estratégia "um-contra-todos"(*One-vs-Rest - OvR*), que quantifica o grau de separabilidade preditiva do modelo, avaliando sua competência em isolar as distribuições matemáticas das diferentes tipologias de intrusão cibernética em relação ao tráfego regular da Internet das Coisas.

$$\text{AUC-ROC}_{\text{Macro}} = \frac{1}{k}\sum_{i=1}^{k} \text{AUC}_i \tag{4.5}$$

onde AUC_i é a área sob a curva ROC da classe i, obtida a partir da Revocação_i (Taxa de Verdadeiros Positivos) em função da Taxa de Falsos Positivos, $\text{TFP}_i = \frac{FP_i}{FP_i + VN_i}$, variando-se o limiar de decisão do classificador.

Para a interpretação das equações estatísticas acima, definem-se as seguintes siglas e variáveis fundamentais derivadas da Matriz de Confusão:

- VP_i **(Verdadeiros Positivos):** Quantidade de fluxos da classe i que foram classificados corretamente pelo modelo preditivo;
- VN_i **(Verdadeiros Negativos):** Quantidade de fluxos pertencentes às demais classes que foram corretamente identificados como não pertencentes à classe i;
- FP_i **(Falsos Positivos):** Quantidade de fluxos de outras categorias que foram incorretamente classificados como sendo da classe i (falso alarme);
- FN_i **(Falsos Negativos):** Quantidade de fluxos pertencentes à classe i que o modelo não conseguiu identificar, rotulando-os erroneamente em outra categoria;
- k **(Número de Classes):** Total de categorias analisadas no problema multiclasse (no contexto deste trabalho, $k = 5$);
- TVP_i e TFP_i: Representam, respectivamente, a **Taxa de Verdadeiros Positivos** (*True Positive Rate*) e a **Taxa de Falsos Positivos** (*False Positive Rate*), calculadas individualmente para a classe i.

4.7 Ranqueamento de *features*

Após o treinamento dos classificadores, o método SHAP foi aplicado individualmente a cada modelo para extrair a importância das variáveis. Para consolidar esses

resultados e definir o subconjunto final de *features*, estruturou-se uma lógica de consenso baseada na contagem de ocorrências e na magnitude do impacto preditivo.

Para cada classificador m (variando de 1 a 5), selecionou-se o conjunto das 15 variáveis com maior importância SHAP, denotado como Top_m. O grau de consenso de uma *feature* j foi calculado pela soma de suas aparições nesses conjuntos:

$$O(j) = \sum_{m=1}^{5} x_{m,j} \tag{4.6}$$

onde $x_{m,j} = 1$ se a característica $j \in \text{Top}_m$, e $x_{m,j} = 0$ caso contrário. Dessa forma, variáveis com valores de $O(j)$ mais altos representam características validadas por um maior número de arquiteturas distintas.

Como critério de desempate para *features* com o mesmo número de ocorrências, calculou-se a magnitude média global da importância SHAP da variável, denotada por $\bar{S}(j)$, considerando apenas os modelos nos quais ela obteve destaque:

$$\bar{S}(j) = \frac{1}{O(j)} \sum_{m=1}^{5} (x_{m,j} \cdot S_{m,j}) \tag{4.7}$$

onde $S_{m,j}$ representa a importância absoluta conferida à *feature* j pelo modelo m.

O ranqueamento final foi estabelecido ordenando-se as variáveis de forma decrescente, primeiramente pelo número de ocorrências $O(j)$ e, em caso de empate, pela magnitude de impacto $\bar{S}(j)$. Esse método simplificado garante uma hierarquia transparente, que prioriza a concordância entre os modelos e preserva o peso matemático das contribuições.

4.8 Treinamento Fase 2

Para cada conjunto de consenso (Top 5, Top 10 e Top 15), com o intuito de refinar a análise e os impactos das mesmas nos classificadores, foi efetuado uma experimentação em níveis, cada um dos classificadores foi treinado novamente com cada um dos cenários de 5, 10 e 15 *features*.

- **Cenário 1 (Espaço Vetorial Integral - Sem SHAP - *Baseline*):** Avaliação de controle submetida à totalidade das *features* da etapa de pré-processamento sem a aplicação de cortes interpretativos, avaliando o impacto das variáveis.
- **Cenário 2 (Top 5 *Features* SHAP):** Subconjunto com as cinco características que exerceram maior impacto preditivo global na decomposição dos modelos: taxa de pacotes (Rate), comprimento do cabeçalho (*Header_Length*), contagem de confirmações (*ack_count*), número de marcações de impulso (*psh_flag_number*) e tempo de vida do pacote (*Time_To_Live*).

- **Cenário 3 (Top 10 *Features* SHAP):** Expansão intermediária: contagem de finalização (*fin_flag_number*), soma total de bytes (*Tot_sum*), média do tamanho dos pacotes (AVG), número de marcações de sincronização (*syn_flag_number*) e o intervalo entre chegadas de pacotes (IAT).
- **Cenário 4 (Top 15 *Features* SHAP):** Subconjunto adicionando métricas de dispersão e controle de protocolo: indicador de criptografia (HTTPS), variância dos comprimentos de pacote (*Variance*), valor máximo numérico (Max), desvio padrão (Std) e contagem de reinicializações (*rst_flag_number*).

4.9 Avaliação - Comparação

Com o fim do treinamento de cada um dos cenários, foi feito o levantamentos dos resultados expositivos em matrizes de confusão e para cada um foi gerado seus resultados de métricas de avaliação (Acurácia, precisão, *recall*, *F1-score*, *AUC-ROC*), o que permitiu uma analise e comparativo dos resultados com o *baseline* onde não teve influencia do SHAP.

5 Experimentos

Neste capítulo, foram avaliados os classificadores Árvore de Decisão, Floresta Aleatória, SVM, AdaBoost e MLP em um cenário Baseline, utilizando as 23 características resultantes do pré-processamento. Em seguida, o método SHAP foi aplicado para interpretar as predições e identificar a importância das características em cada modelo. Com base nesses resultados, foi elaborado um ranqueamento por consenso para definir os conjuntos Top 5, Top 10 e Top 15, utilizados em novos treinamentos sob as mesmas condições experimentais. Por fim, os desempenhos foram comparados por meio das matrizes de confusão e das métricas Acurácia, Precisão Macro, Revocação Macro, F1-Score Macro e AUC-ROC Macro, permitindo avaliar o impacto da seleção de características em cada classificador.

5.1 *Baseline*

O experimento *Baseline* representa o cenário de referência da pesquisa: os cinco classificadores foram treinados sobre o espaço integral de *features* resultante da Etapa 2 de pré-processamento (Seção 4.2), ou seja, sobre as 23 *features* remanescentes após a remoção de colunas constantes, duplicadas e altamente correlacionadas, sem qualquer redução adicional de dimensionalidade via SHAP.

Tabela 3 – Características (*features*) utilizadas no *Baseline* após o pré-processamento

Feature	**Descrição**	**Justificativa de Permanência**
Header_Length	Comprimento do cabeçalho do pacote de rede.	Passou no filtro de variância e manteve correlação de Pearson inferior a 0,80 com outras métricas.
Time_To_Live	Tempo de vida do pacote (TTL) na rede.	Apresentou alta dispersão (útil para detectar o perfil do dispositivo) e evadiu os cortes de colinearidade.
Rate	Taxa de transmissão de pacotes por segundo.	Métrica volumétrica primária com alta entropia; superou a validação de independência linear.
fin_flag_number	Contagem de pacotes com a *flag* FIN.	Aprovada no filtro de variância não-nula e sem forte redundância estatística com outras marcações TCP.
syn_flag_number	Contagem de pacotes com a *flag* SYN.	Possui alta dispersão seletiva (importante para detectar *SYN Flood*); correlação mantida abaixo de 0,80.
rst_flag_number	Contagem de pacotes com a *flag* RST.	Comportamento estatístico independente; não foi eliminada pelo filtro de remoção de duplicatas.
psh_flag_number	Contagem de pacotes com a *flag* PSH.	Retida pelo limiar de variância e validada pelo filtro rigoroso de colinearidade da etapa 2.
ece_flag_number	Contagem de pacotes com a *flag* ECE.	Variância válida constatada; ausência de duplicação exata com outras colunas de *flags* no *dataset*.

Continua na próxima página...

Tabela 3 – Continuação

Feature	**Descrição**	**Justificativa de Permanência**
ack_count	Contagem da *flag* de confirmação ACK.	Variável não constante; preservada pela matriz de correlação cruzada de Pearson.
HTTPS	Tráfego associado ao protocolo seguro HTTPS.	Atributo binário aprovado no limite de dependência linear predefinido ($< 0,80$).
DNS	Tráfego de resolução de nomes DNS.	Apresentou variância útil para a classificação; comportamento matemático independente de outros protocolos.
SSH	Tráfego do protocolo de acesso remoto SSH.	Sobreviveu à eliminação de variáveis por não possuir correlação linear forte com outras portas lógicas.
IRC	Tráfego do protocolo de comunicação IRC.	Evadiu a remoção estatística automática, demonstrando contribuição de dispersão própria.
DHCP	Tráfego de configuração dinâmica DHCP.	Estatisticamente independente; aprovada no filtro automático de variância e redundância.
ICMP	Tráfego de mensagens de erro ICMP.	Essencial para detecção de anomalias volumétricas; não apresentou redundância com métricas TCP/UDP.
IGMP	Tráfego de grupos *multicast* IGMP.	Variância constatada como válida e baixa correlação absoluta no mapeamento do conjunto de dados.
IPv	Tráfego geral do protocolo de internet IP.	Transpôs o filtro de colunas duplicadas e o corte rigoroso de multicolinearidade.

Continua na próxima página...

Tabela 3 – Continuação

Feature	**Descrição**	**Justificativa de Permanência**
Tot_sum	Soma total de *bytes* transmitidos no fluxo.	Métrica não constante e sem redundância linear direta ($< 0,80$) com as variáveis médias de tamanho.
Max	Tamanho máximo do pacote registrado.	Aprovada na etapa de descorrelação por manter independência estatística útil ao modelo.
AVG	Média de tamanho dos pacotes.	O coeficiente de Pearson em relação a variáveis semelhantes manteve-se estritamente abaixo do corte de 80%.
Std	Desvio padrão numérico do tamanho dos pacotes.	Retida pelos filtros programados por aportar um cálculo independente de dispersão aos classificadores.
IAT	Tempo entre chegadas (*Inter-Arrival Time*).	Variável temporal de alta utilidade discriminatória; evadiu todos os cortes prévios programados.
Variance	Variância dos comprimentos de pacote.	Validada automaticamente pelo algoritmo por não apresentar valores constantes ou redundância de 100%.

Fonte: elaborado pelo autor.

Esse cenário serve como ponto de comparação para os experimentos subsequentes com seleção de *features* (Seção 5.3), permitindo isolar o efeito da redução de *features* sobre o desempenho de cada modelo.

O conjunto de dados foi dividido uma única vez em treino e teste, na proporção 80/20, de forma estratificada em relação à classe, e cada classificador foi treinado individualmente com os hiperparâmetros apresentados na Tabela 2, mantidos idênticos em todos os cenários experimentais do trabalho (*Baseline* e Top 5, 10 e 15 SHAP), de modo que eventuais variações de desempenho entre cenários possam ser atribuídas exclusivamente à alteração do espaço de *features*, e não a mudanças na configuração dos modelos. Para

cada classificador, foram computadas as cinco métricas multiclasse descritas na Seção 4.6 (Acurácia, Precisão Macro, Revocação Macro, F1-Score Macro e AUC-ROC Macro) e foi gerada a respectiva Matriz de Confusão, apresentadas individualmente nas subseções a seguir.

5.1.1 Árvore Decisão (resultados)

No experimento com a Árvore de Decisão, o modelo foi instanciado com os hiperparâmetros `random_state=42` e `class_weight='balanced'`, mantendo o critério de divisão de nós `gini` em seu valor padrão, conforme especificado na Tabela 2 e alinhado à caracterização apresentada na Seção 2.4.2, que descreve o algoritmo como um modelo simbólico estruturado em divisões hierárquicas do espaço de *features*. O classificador foi treinado sobre a partição de treino do espaço integral de *features* (Seção 4.2) e avaliado sobre a partição de teste reservada, gerando a matriz de confusão da Figura 2 e as métricas multiclasse consolidadas na Tabela 4, calculadas conforme os critérios descritos na Seção 4.6.

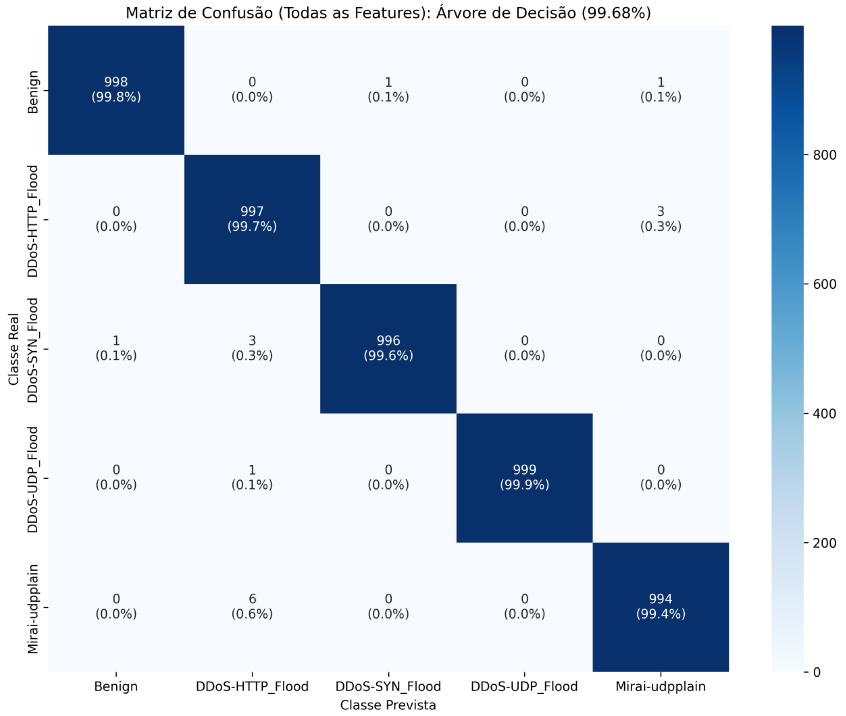

Figura 2 – Matriz de Confusão Arvore de Decisão - *Baseline*.

Fonte: elaborado pelo autor.

Tabela 4 – Resultados obtidos pelo modelo Árvore de Decisão

Métrica	**Valor**
Acurácia	0.9968
Precisão Macro	0.9968
Revocação Macro	0.9968
F1-Score Macro	0.9968
AUC-ROC Macro	0.9980

Fonte: elaborado pelo autor.

5.1.2 *Random Forest* (Resultados)

No experimento com a Floresta Aleatória, o modelo foi configurado com `n_estimators=100`, `random_state=42`, `class_weight='balanced'` e paralelização via `n_jobs=-1`, conforme a Tabela 2, refletindo a arquitetura de *ensemble* baseada em *bagging* descrita na Seção 2.4.2, na qual cem árvores de decisão independentes são treinadas sobre subconjuntos aleatórios de amostras e *features*. O treinamento seguiu a mesma partição de treino utilizada pelos demais classificadores do cenário *Baseline*, e a avaliação sobre a partição de teste produziu a matriz de confusão da Figura 3 e as métricas apresentadas na Tabela 5, seguindo os mesmos critérios de cálculo definidos na Seção 4.6.

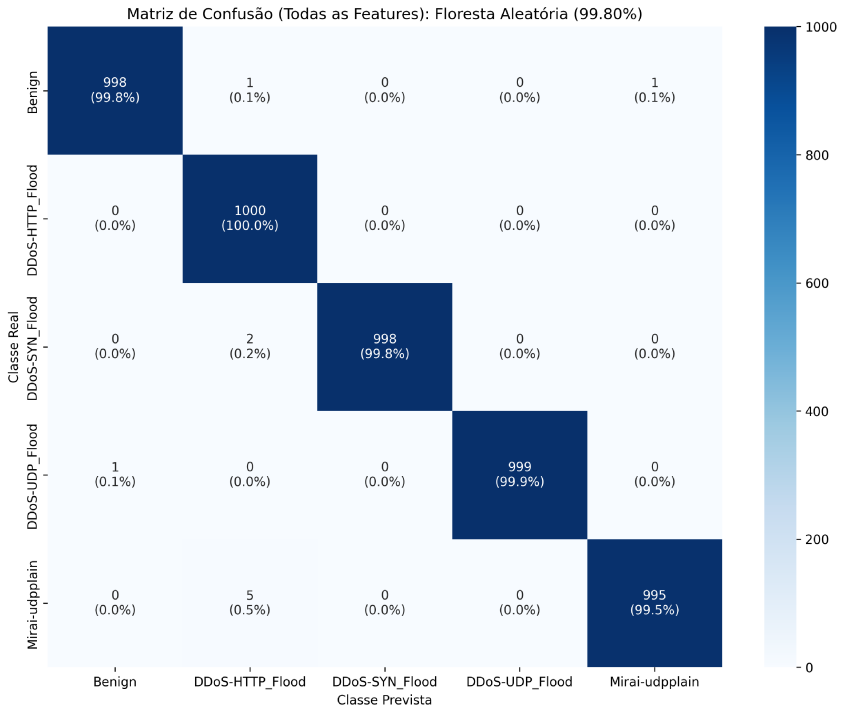

Figura 3 – Matriz de Confusão Floresta Aleatória - *Baseline*.

Fonte: elaborado pelo autor.

Tabela 5 – Resultados obtidos pelo modelo Floresta Aleatória

Métrica	**Valor**
Acurácia	0.9980
Precisão Macro	0.9980
Revocação Macro	0.9980
F1-Score Macro	0.9980
AUC-ROC Macro	1.0000

Fonte: elaborado pelo autor.

5.1.3 *SVM* (Resultados)

No experimento com a SVM, o classificador foi configurado com `kernel='rbf'` (padrão), `probability=True`, `random_state=42` e `class_weight='balanced'`, conforme a Tabela 2, em linha com a descrição do algoritmo na Seção 2.4.2 como um classificador discriminativo que mapeia os dados em espaços de dimensão superior por meio de funções de *kernel* para traçar hiperplanos de margem máxima. O treinamento e a avaliação seguiram o mesmo procedimento aplicado aos demais modelos do cenário *Baseline*, sobre o espaço integral de *features*, resultando na matriz de confusão da Figura 4 e nas métricas consolidadas na Tabela 6, calculadas conforme os critérios da Seção 4.6.

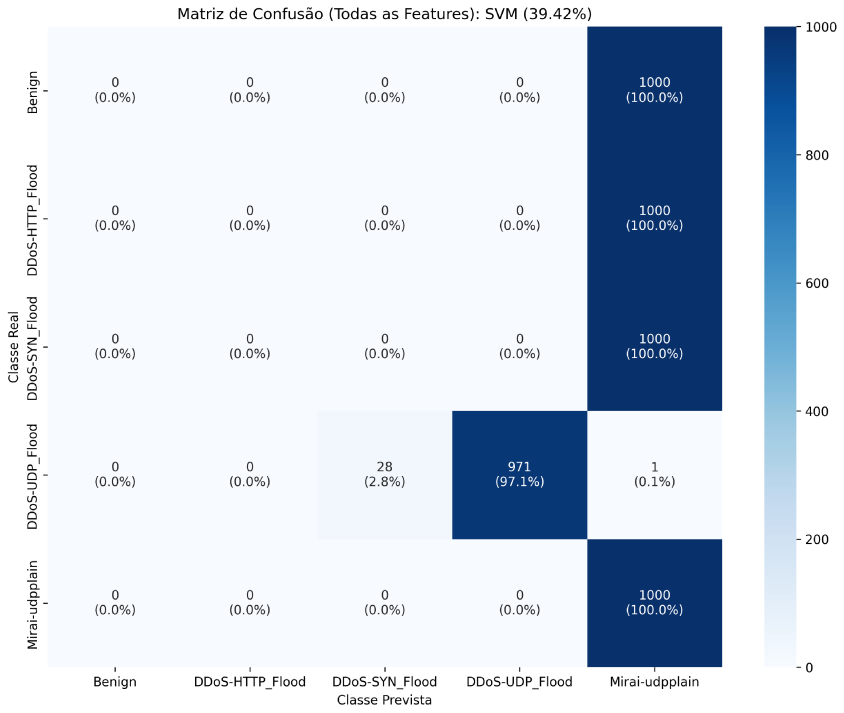

Figura 4 – Matriz de Confusão SVM - *Baseline*.

Fonte: elaborado pelo autor.

Tabela 6 – Resultados obtidos pelo modelo SVM

Métrica	**Valor**
Acurácia	0.3942
Precisão Macro	0.2500
Revocação Macro	0.3942
F1-Score Macro	0.2770
AUC-ROC Macro	0.7000

Fonte: elaborado pelo autor.

5.1.4 *Adaboost* (Resultados)

No experimento com o AdaBoost, o modelo foi configurado com `n_estimators=100` e `random_state=42`, conforme a Tabela 2, sem o parâmetro `class_weight`, diferentemente dos demais classificadores do estudo. A configuração reflete o mecanismo de *boosting* sequencial descrito na Seção 2.4.2, no qual classificadores fracos são treinados iterativamente, com reponderação progressiva das instâncias classificadas incorretamente a cada rodada. O treinamento seguiu a mesma partição de treino e teste utilizada pelos demais modelos do cenário *Baseline*, produzindo a matriz de confusão da Figura 5 e as métricas apresentadas na Tabela 7, conforme os critérios definidos na Seção 4.6.

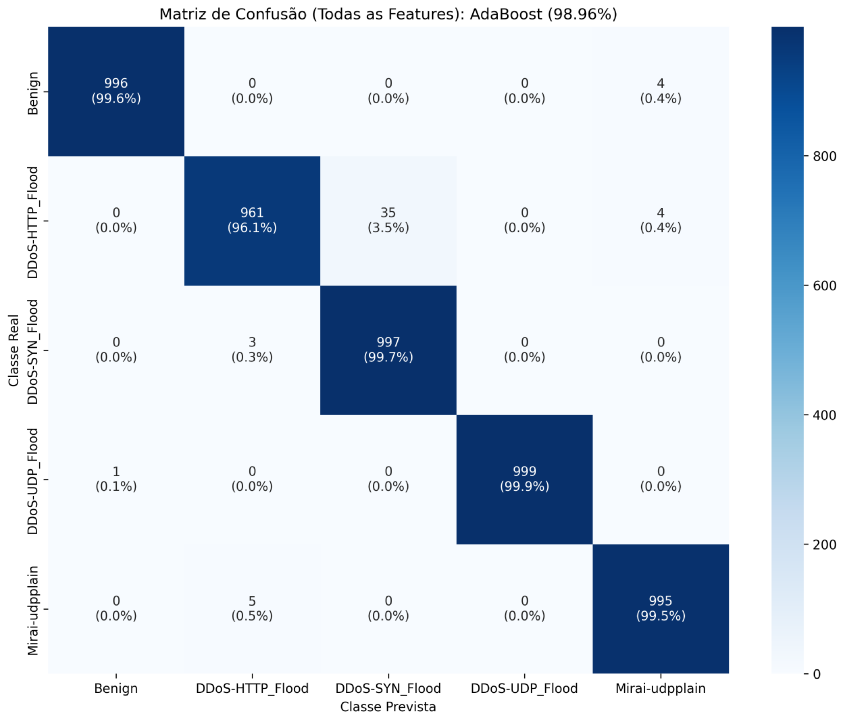

Figura 5 – Matriz de Confusão Adaboost - *Baseline*.

Fonte: elaborado pelo autor.

Tabela 7 – Resultados obtidos pelo modelo AdaBoost

Métrica	**Valor**
Acurácia	0.9896
Precisão Macro	0.9898
Revocação Macro	0.9896
F1-Score Macro	0.9896
AUC-ROC Macro	0.9995

Fonte: elaborado pelo autor.

5.1.5 MLP (resultados)

No experimento com a Rede Neural (MLP), o modelo foi configurado com três camadas ocultas de 128, 64 e 32 neurônios (`hidden_layer_sizes=(128,64,32)`), função de ativação `relu`, otimizador `adam`, `max_iter=100`, `early_stopping=True` e `random_state=42`, conforme a Tabela 2, alinhado à descrição da arquitetura de aprendizado profundo apresentada na Seção 2.4.2. O classificador foi treinado sobre a mesma partição de treino reservada para os demais modelos do cenário *Baseline* e avaliado sobre a partição de teste, gerando a matriz de confusão da Figura 6 e as métricas consolidadas na Tabela 8, calculadas conforme os critérios descritos na Seção 4.6.

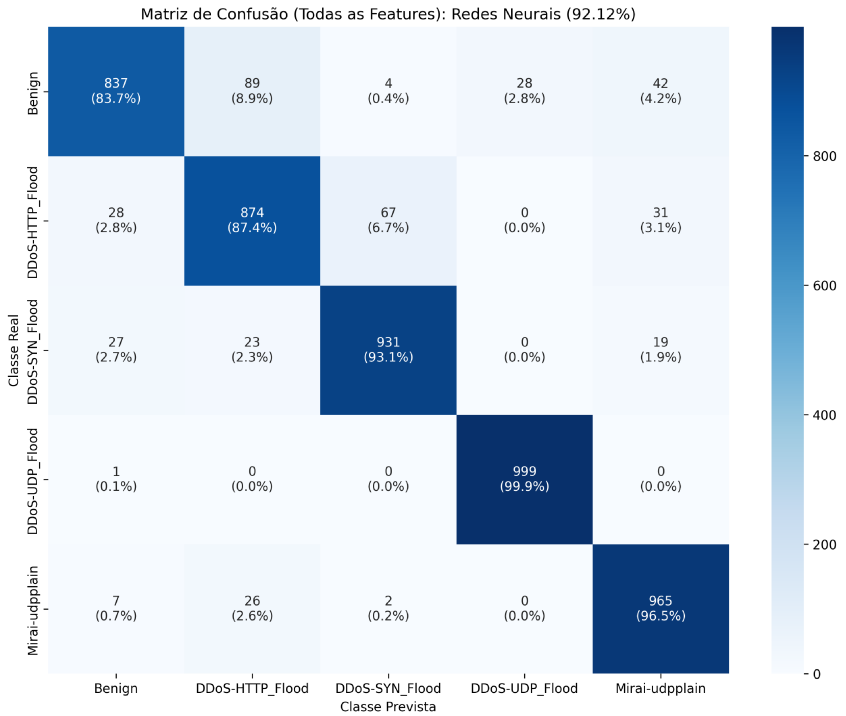

Figura 6 – Matriz de Confusão MLP - *Baseline*.

Fonte: elaborado pelo autor.

Tabela 8 – Resultados obtidos pelo modelo Redes Neurais (MLP)

Métrica	**Valor**
Acurácia	0.9212
Precisão Macro	0.9213
Revocação Macro	0.9212
F1-Score Macro	0.9206
AUC-ROC Macro	0.9732

Fonte: elaborado pelo autor.

5.2 Explicabilidade com SHAP

Nesta etapa dos experimentos, o método SHAP foi aplicado aos cinco classificadores treinados no cenário Baseline para interpretar suas predições e identificar a contribuição de cada feature. Para a Árvore de Decisão e a Floresta Aleatória, utilizou-se o TreeExplainer, enquanto, para a SVM, o AdaBoost e a MLP, foi empregado o KernelExplainer. Com base nos valores de SHAP, calculou-se a importância média das características em cada modelo, o que permitiu gerar os gráficos de resumo e construir o ranqueamento por consenso utilizado na definição dos conjuntos Top 5, Top 10 e Top 15.

5.2.1 Árvore de Decisão (resultados SHAP)

No experimento de explicabilidade da Árvore de Decisão, aplicou-se o algoritmo *TreeExplainer* (Seção 2.6), que explora diretamente a estrutura de nós e ramificações do modelo para calcular os valores de *Shapley* de forma exata sobre o conjunto de teste. Para cada *feature* de rede, calculou-se a importância média segundo a Equação 4.7, e os resultados foram sumarizados no gráfico de resumo SHAP apresentado na Figura 7.

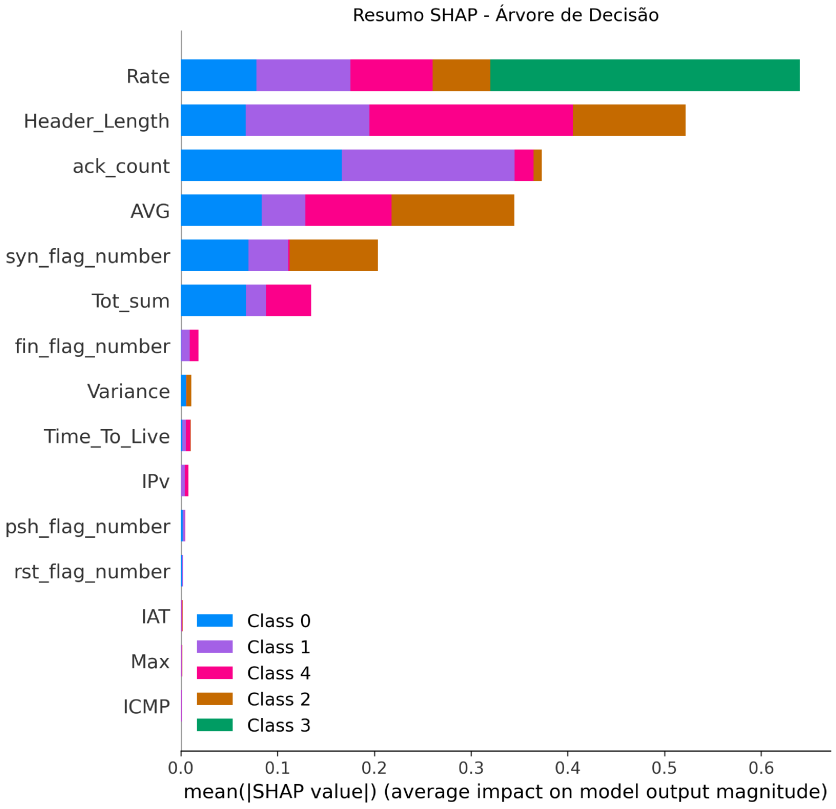

Figura 7 – Árvore de Decisão - *SHAP*.

Fonte: elaborado pelo autor.

5.2.2 *Random Forest* (resultados *SHAP*)

No experimento de explicabilidade da Floresta Aleatória, o cálculo dos valores de *Shapley* também foi realizado por meio do *TreeExplainer* (Seção 2.6), estendendo o mesmo princípio de exploração da estrutura de árvores ao conjunto de estimadores que compõem o *ensemble*. A importância média de cada *feature* foi obtida conforme a Equação 4.7, com o resultado consolidado apresentado no gráfico de resumo SHAP da Figura 8.

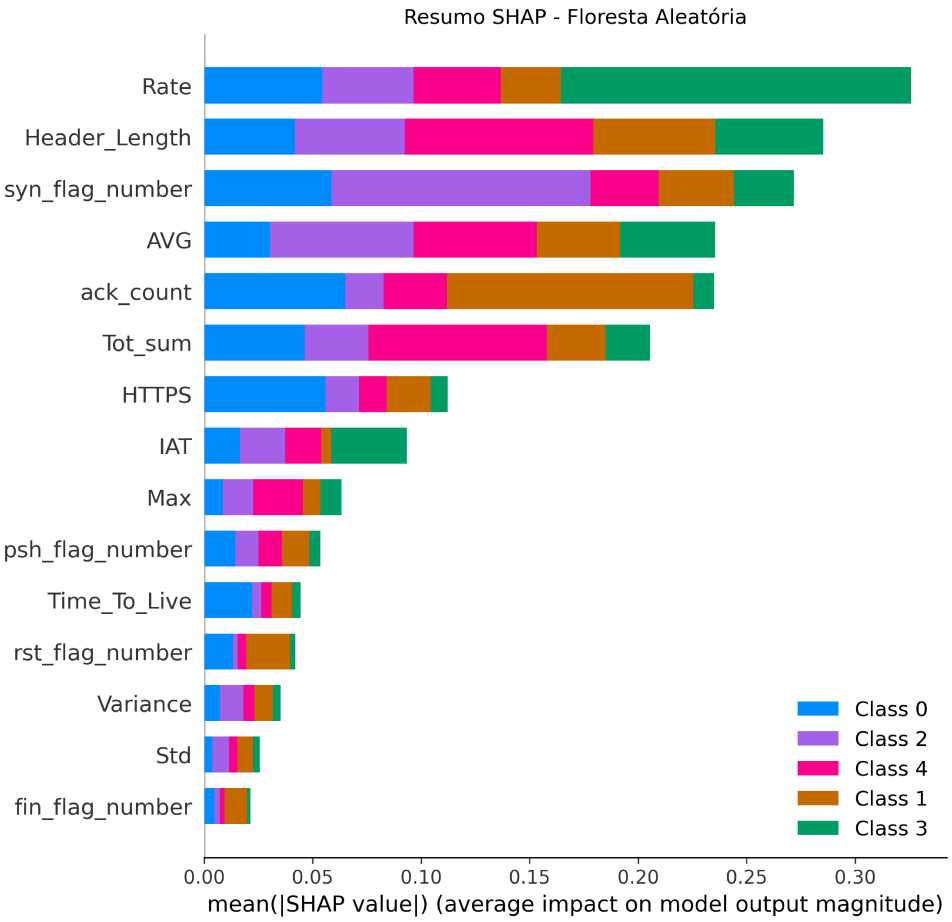

Figura 8 – Floresta Aleatória - *SHAP*.

Fonte: elaborado pelo autor.

5.2.3 SVM (resultados *SHAP*)

No experimento de explicabilidade da SVM, empregou-se o *KernelExplainer* (Seção 2.6), abordagem agnóstica ao modelo utilizada nos classificadores sem estrutura de árvore, aplicada sobre uma amostra representativa e estratificada por classe do conjunto de teste. A importância média de cada *feature* foi calculada conforme a Equação 4.7, e os resultados foram sumarizados no gráfico de resumo SHAP apresentado na Figura 9.

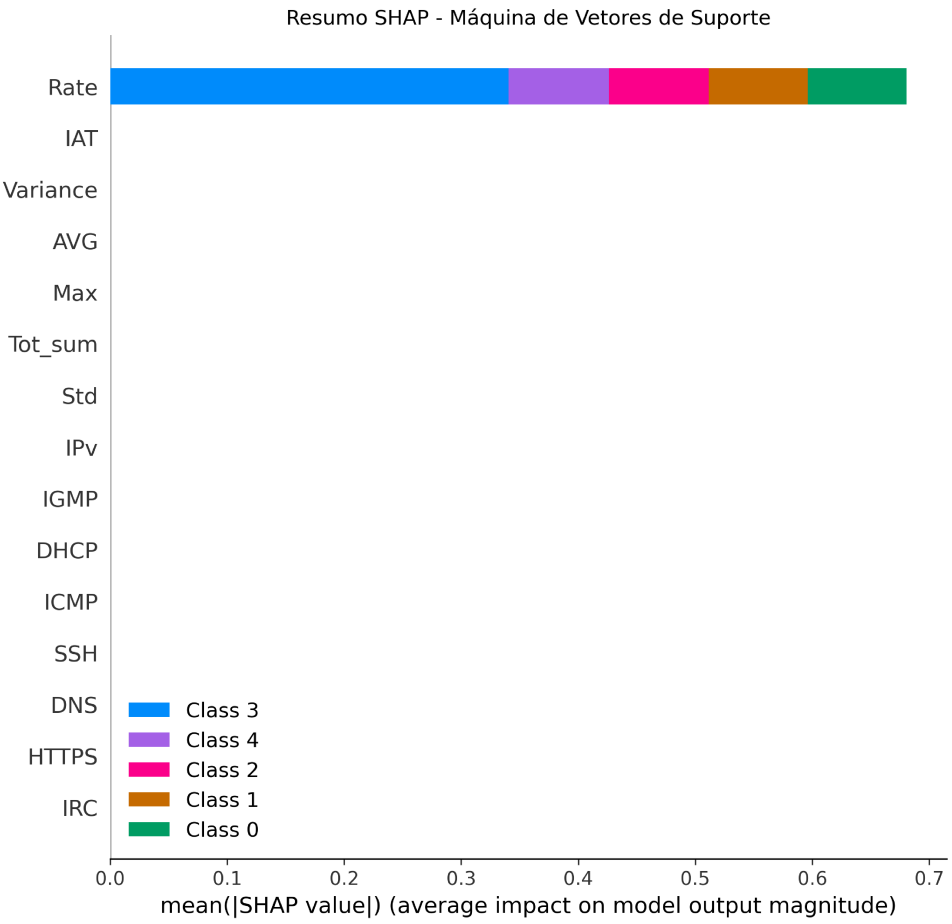

Figura 9 – SVM - *SHAP*.

Fonte: elaborado pelo autor.

5.2.4 *Adaboost* (resultados SHAP)

No experimento de explicabilidade do AdaBoost, os valores de *Shapley* também foram estimados por meio do *KernelExplainer* (Seção 2.6), aplicado sobre uma amostra representativa e estratificada por classe do conjunto de teste. A importância média de cada *feature* seguiu a mesma formulação da Equação 4.7, com os resultados sumarizados no gráfico de resumo SHAP apresentado na Figura 10.

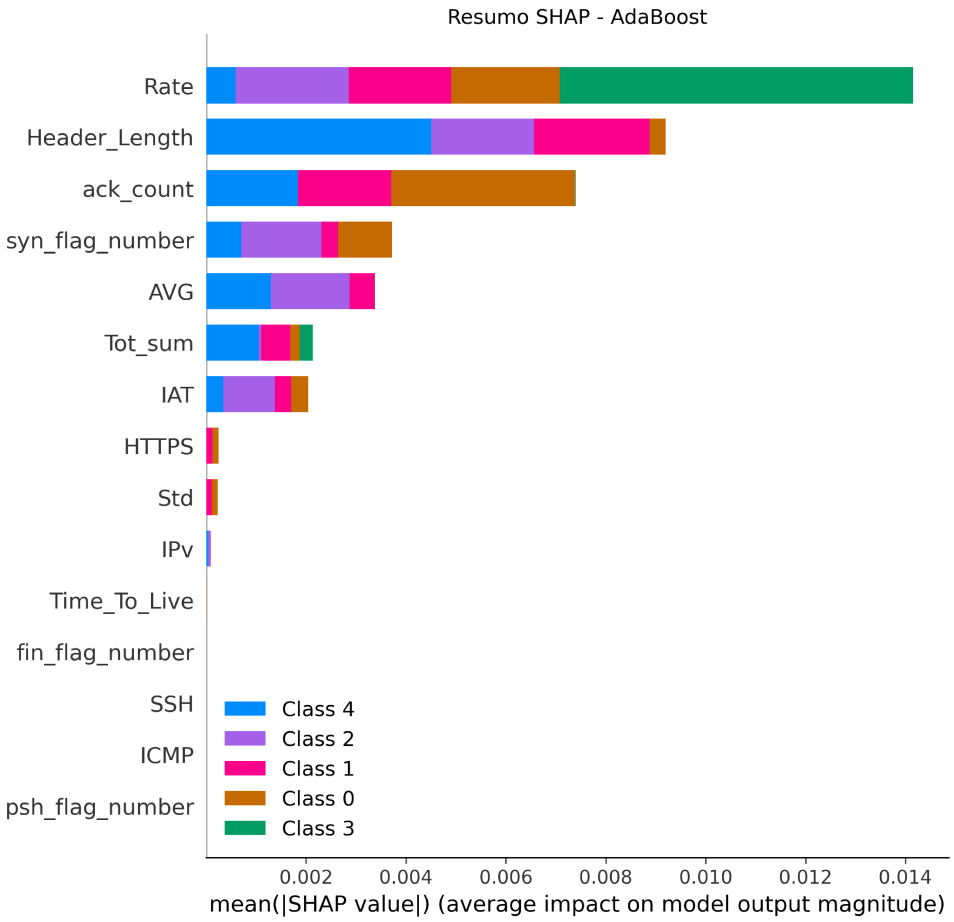

Figura 10 – *Adaboost - SHAP.*

Fonte: elaborado pelo autor.

5.2.5 MLP (resultados SHAP)

No experimento de explicabilidade da Rede Neural (MLP), o cálculo dos valores de *Shapley* seguiu o mesmo procedimento aplicado à SVM e ao AdaBoost, com o *KernelExplainer* (Seção 2.6) operando sobre uma amostra representativa e estratificada por classe do conjunto de teste. A importância média de cada *feature* foi calculada conforme a Equação 4.7, e os resultados foram sumarizados no gráfico de resumo SHAP apresentado na Figura 11.

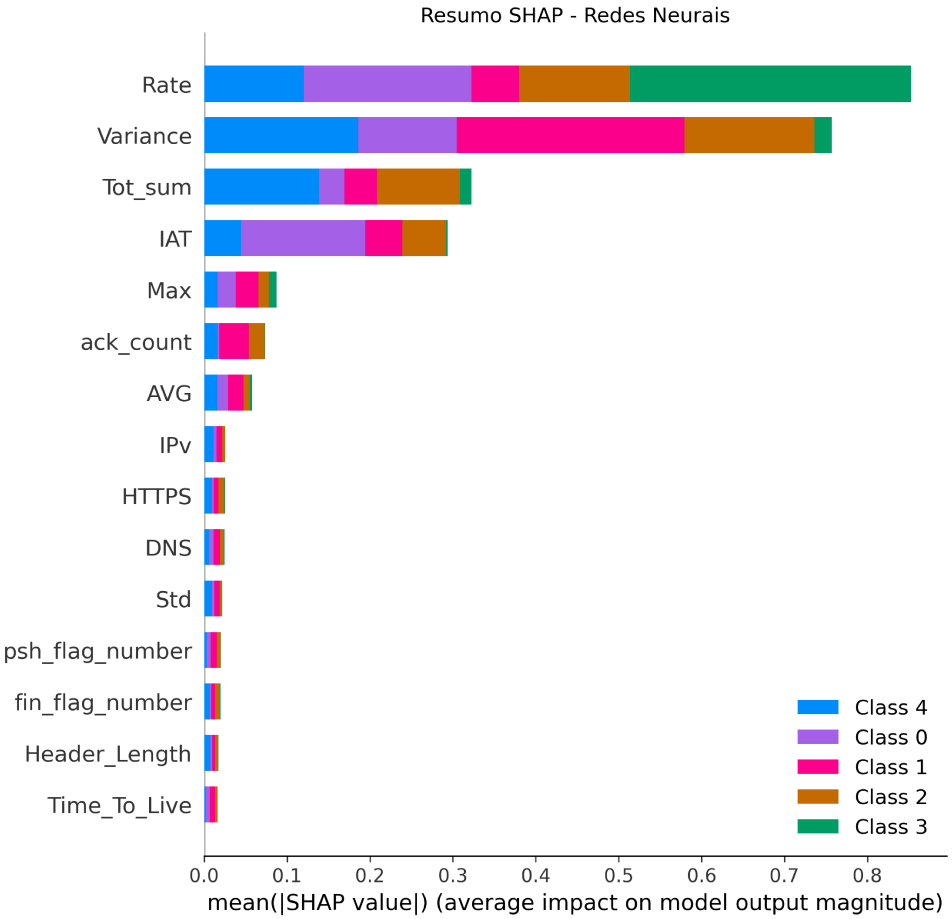

Figura 11 – *mlp - SHAP.*

Fonte: elaborado pelo autor.

5.3 Consenso e Ranqueamento

Nesta etapa, os resultados de importância obtidos com o SHAP para os cinco classificadores foram consolidados em um ranqueamento por consenso. As *features* foram ordenadas considerando sua recorrência entre os modelos e, como critério complementar, sua importância média global de SHAP. A partir desse ranqueamento, foram definidos os conjuntos Top 5, Top 10 e Top 15, posteriormente utilizados nos novos treinamentos para avaliar o impacto da redução de dimensionalidade no desempenho dos classificadores. A Figura 12 ilustra a organização de 15 das 23 *features* utilizadas no *baseline*.

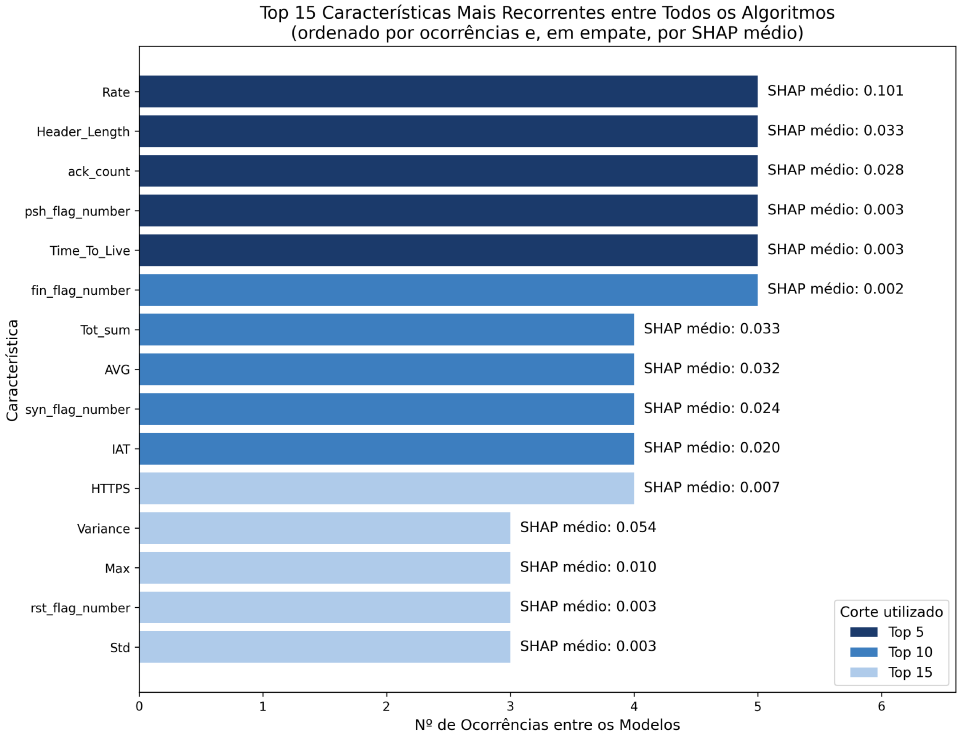

Figura 12 – Top 15 - *SHAP*.

Fonte: elaborado pelo autor.

As tabelas 9, 10 e 11 apresentam os subconjuntos de features extraídos a partir do consenso e do ranqueamento do SHAP. Para cada característica, são indicados o número de ocorrências entre os cinco classificadores e sua importância média global de SHAP, permitindo identificar as variáveis mais relevantes e recorrentes. Esses subconjuntos foram utilizados nos experimentos seguintes para verificar como diferentes níveis de redução de dimensionalidade afetam o desempenho dos modelos.

Tabela 9 – Top 5 Características Mais Recorrentes

Característica	**Nº de Ocorrências**	**SHAP Médio Global**
Rate	5	0.101
Header_Length	5	0.033
ack_count	5	0.028
psh_flag_number	5	0.003
Time_To_Live	5	0.003

Fonte: elaborado pelo autor.

Tabela 10 – Top 10 Características Mais Recorrentes

Característica	**Nº de Ocorrências**	**SHAP Médio Global**
Rate	5	0.101
Header_Length	5	0.033
ack_count	5	0.028
psh_flag_number	5	0.003
Time_To_Live	5	0.003
fin_flag_number	5	0.002
Tot_sum	4	0.033
AVG	4	0.032
syn_flag_number	4	0.024
IAT	4	0.020

Fonte: elaborado pelo autor.

Tabela 11 – Top 15 Características Mais Recorrentes

Característica	**Nº de Ocorrências**	**SHAP Médio Global**
Rate	5	0.101
Header_Length	5	0.033
ack_count	5	0.028
psh_flag_number	5	0.003
Time_To_Live	5	0.003
fin_flag_number	5	0.002
Tot_sum	4	0.033
AVG	4	0.032
syn_flag_number	4	0.024
IAT	4	0.020
HTTPS	4	0.007
Variance	3	0.054
Max	3	0.010
rst_flag_number	3	0.003
Std	3	0.003

Fonte: elaborado pelo autor.

5.4 Treinamento Baseado no Consenso

Nesta etapa, os cinco classificadores foram novamente treinados com os conjuntos Top 5, Top 10 e Top 15 definidos pelo consenso e pelo ranqueamento do SHAP. Em todos os cenários, foram mantidos os mesmos parâmetros dos modelos, a mesma divisão estratificada dos dados e as mesmas métricas de avaliação empregadas no *baseline*. Esse procedimento permitiu comparar os resultados e avaliar objetivamente o impacto da seleção de *features* no desempenho de cada modelo.

5.4.1 Conjunto - Top 5 *features*

Nesta primeira etapa experimental, os cinco classificadores de aprendizado de máquina foram submetidos ao treinamento e validação utilizando apenas o subconjunto das 5 características mais relevantes ranqueadas pelo método SHAP (Rate, Header_Length, ack_count, psh_flag_number e Time_To_Live). Ao final do treinamento, cada classificador e o codificador de rótulos associado foram serializados (`joblib`) e armazenados individualmente.

5.4.1.1 Árvore Decisão

No experimento com a Árvore de Decisão neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 5 do ranqueamento SHAP descrito no início desta subseção. O treinamento e a avaliação seguiram a mesma metodologia de divisão estratificada 80/20 empregada no cenário *Baseline*, resultando na matriz de confusão apresentada na Figura 13, com as métricas correspondentes consolidadas na Tabela 12.

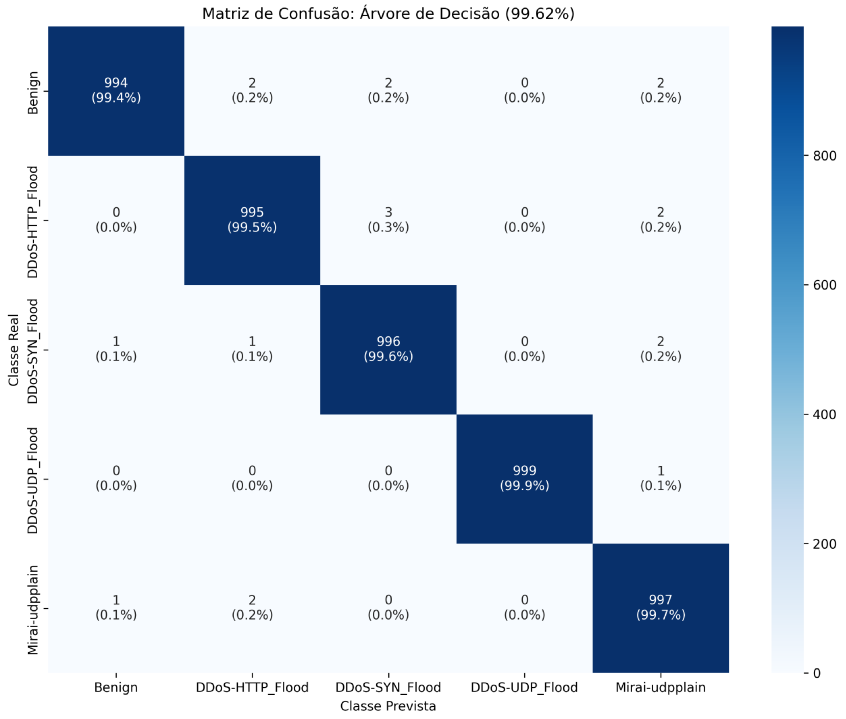

Figura 13 – Top 5 - ad - *SHAP*.

Fonte: elaborado pelo autor.

5.4.1.2 *Random Forest*

No experimento com a Floresta Aleatória neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 5 do ranqueamento SHAP. O treinamento e a avaliação seguiram a mesma partição estratificada 80/20 utilizada nos demais cenários, resultando na matriz de confusão apresentada na Figura 14, com as métricas correspondentes consolidadas na Tabela 12.

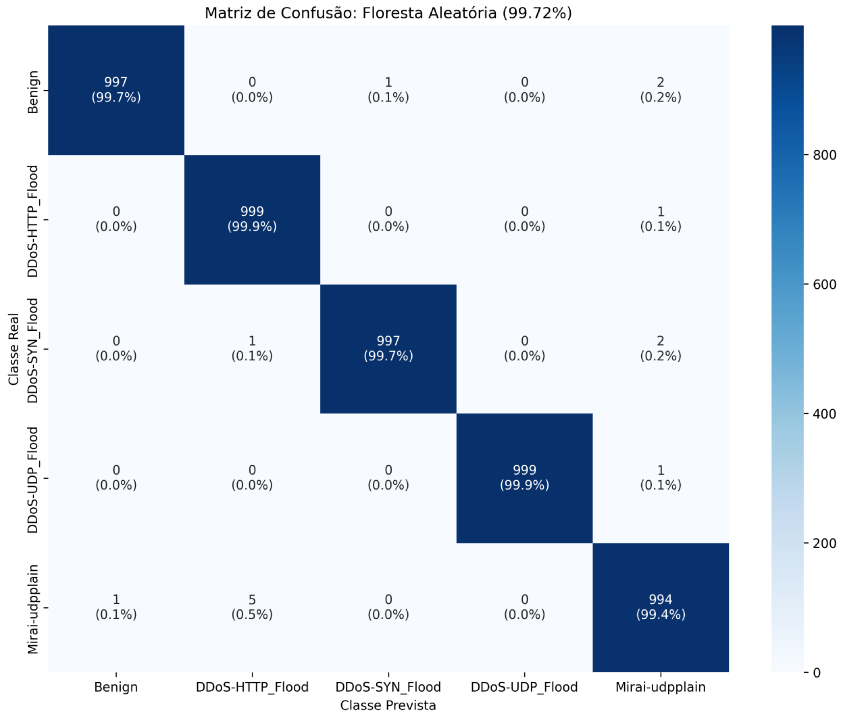

Figura 14 – Top 1 - rf - *SHAP*.

Fonte: elaborado pelo autor.

5.4.1.3 *SVM*

No experimento com a SVM neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 5 do ranqueamento SHAP. O treinamento e a avaliação seguiram a mesma partição estratificada 80/20 utilizada nos demais cenários, resultando na matriz de confusão apresentada na Figura 15, com as métricas correspondentes consolidadas na Tabela 12.

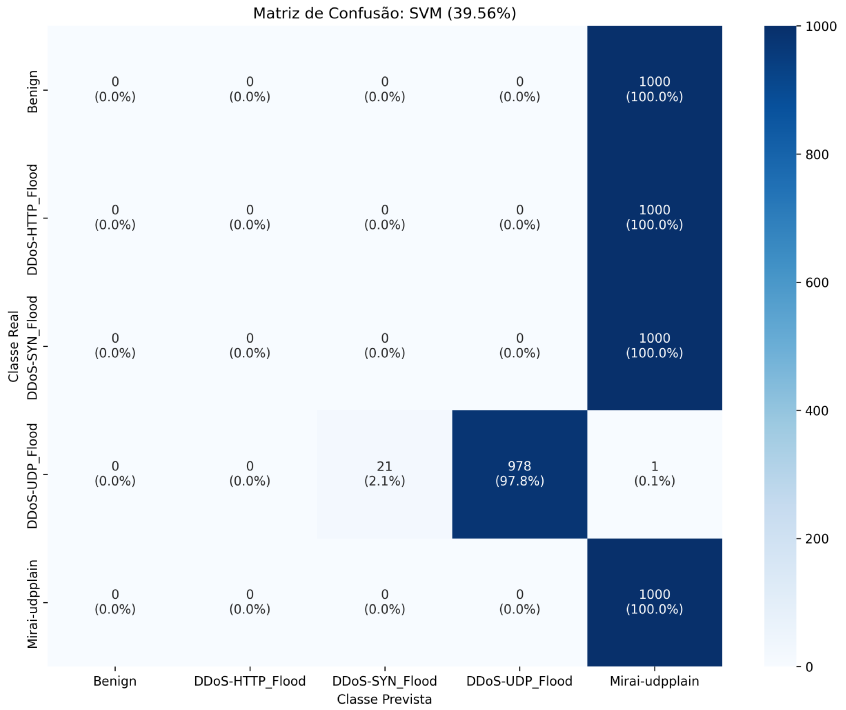

Figura 15 – Top 5 - SVM - *SHAP*.

Fonte: elaborado pelo autor.

5.4.1.4 *Adaboost*

No experimento com o AdaBoost neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 5 do ranqueamento SHAP. O treinamento e a avaliação seguiram a mesma partição estratificada 80/20 utilizada nos demais cenários, resultando na matriz de confusão apresentada na Figura 16, com as métricas correspondentes consolidadas na Tabela 12.

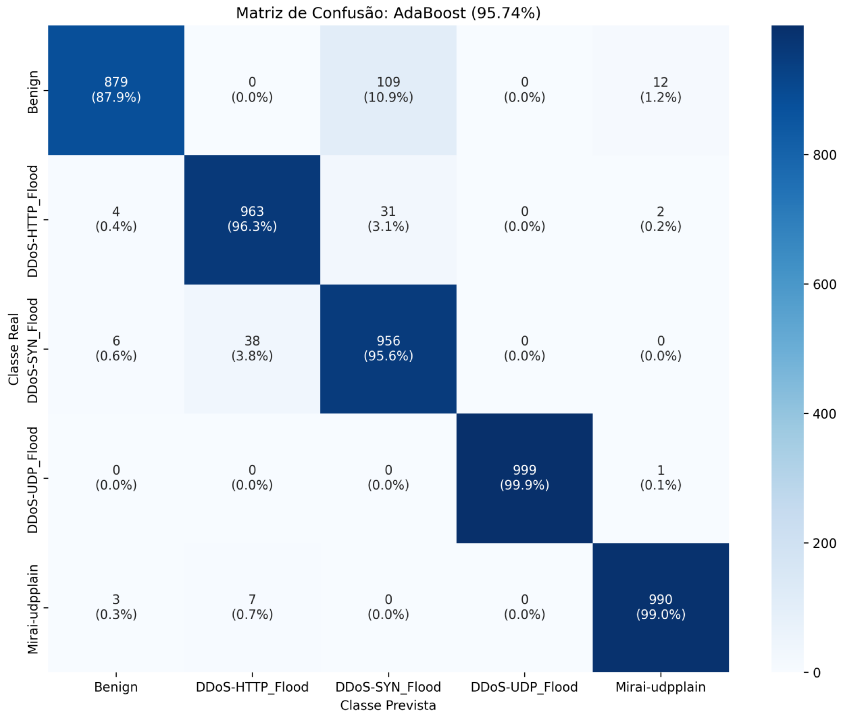

Figura 16 – Top 5 - adaboost *SHAP*.

Fonte: elaborado pelo autor.

5.4.1.5 *MLP*

No experimento com a Rede Neural (MLP) neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 5 do ranqueamento SHAP. O treinamento e a avaliação seguiram a mesma partição estratificada 80/20 utilizada nos demais cenários, resultando na matriz de confusão apresentada na Figura 17, com as métricas correspondentes consolidadas na Tabela 12.

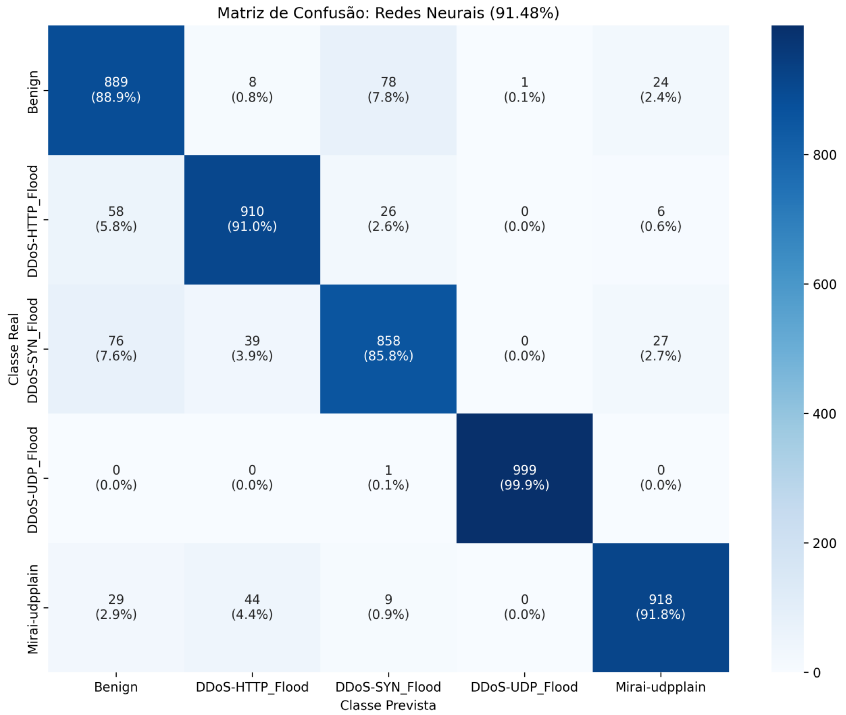

Figura 17 – Top 5 - MLP - *SHAP*.

Fonte: elaborado pelo autor.

5.4.1.6 Resultados Consolidados - Top 5 *features* - SHAP

A Tabela 12 consolida o desempenho geral das métricas avaliadas para todos os algoritmos neste cenário restrito de *features*.

Tabela 12 – Consolidação das Métricas de Desempenho — Cenário Top 5 *Features SHAP*.

Modelo	**Acurácia**	**Precisão**	***Recall***	***F1-Score***	***AUC-ROC***
Árvore de Decisão	0.9962	0.9962	0.9962	0.9962	0.9976
Floresta Aleatória	0.9972	0.9972	0.9972	0.9972	0.9993
SVM	0.3956	0.2500	0.3956	0.2778	0.7000
AdaBoost	0.9574	0.9596	0.9574	0.9575	0.9968
Redes Neurais	0.9148	0.9155	0.9148	0.9150	0.9763

Fonte: elaborado pelo autor.

5.4.2 Conjunto - Top 10 *features*

Na segunda etapa experimental, o espaço dimensional foi expandido para incluir as 10 características mais relevantes, ranqueadas pelo método de interpretabilidade SHAP. A este subconjunto, que já contava com *Rate, Header_Length, ack_count, psh_flag_number e Time_To_Live*, foram adicionadas variáveis estatísticas e temporais críticas do tráfego de rede: contagem de finalização de conexão (*fin_flag_number*), soma total de bytes (*Tot_sum*), média do tamanho dos pacotes (*AVG*), contagem de requisições de sincronização (*syn_flag_number*) e o tempo entre chegadas de pacotes (Inter-Arrival Time(IAT)).

5.4.2.1 Árvore Decisão

No experimento com a Árvore de Decisão neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 10 do ranqueamento SHAP descrito no início desta subseção. O treinamento e a avaliação seguiram a mesma metodologia de divisão estratificada 80/20 empregada nos demais cenários, resultando na matriz de confusão apresentada na Figura 18, com as métricas correspondentes consolidadas na Tabela 13.

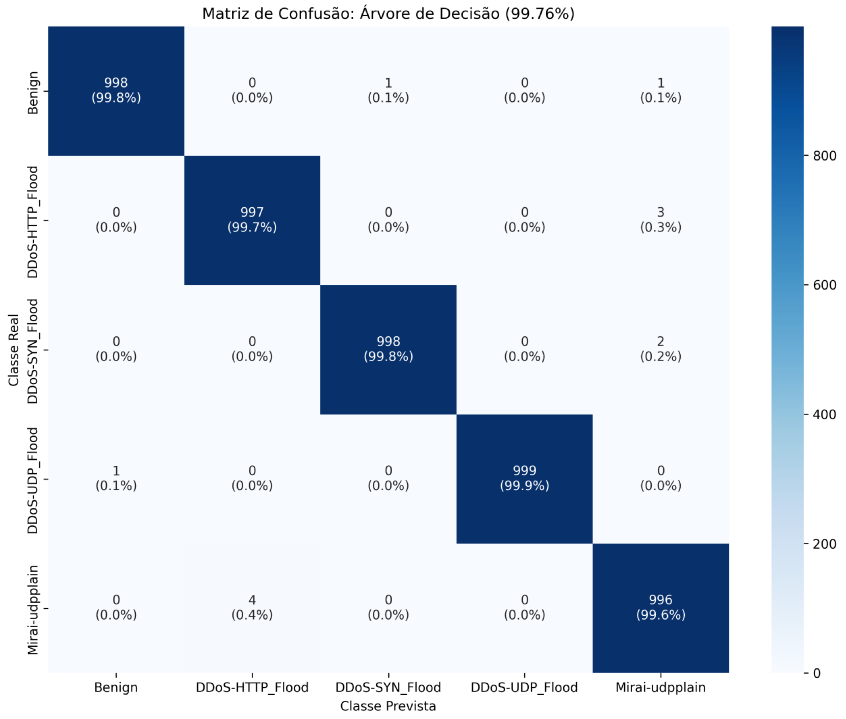

Figura 18 – Top 10 - ad - *SHAP*.

Fonte: elaborado pelo autor.

5.4.2.2 Random Forest

No experimento com a Floresta Aleatória neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 10 do ranqueamento SHAP. O treinamento e a avaliação seguiram a mesma partição estratificada 80/20 utilizada nos demais cenários, resultando na matriz de confusão apresentada na Figura 19, com as métricas correspondentes consolidadas na Tabela 13.

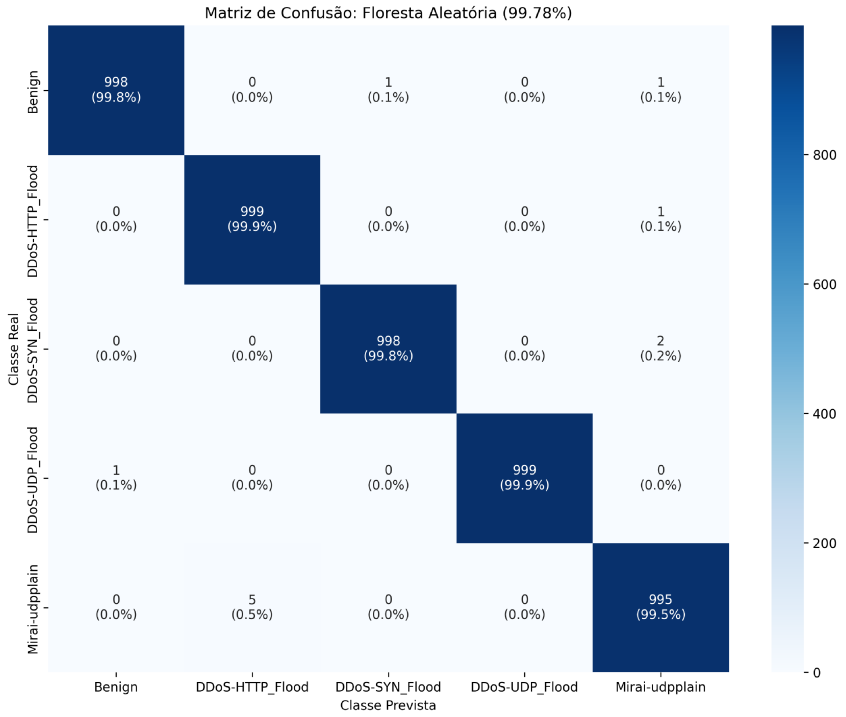

Figura 19 – Top 10 - rf - *SHAP*.

Fonte: elaborado pelo autor.

5.4.2.3 SVM

No experimento com a SVM neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 10 do ranqueamento SHAP. O treinamento e a avaliação seguiram a mesma partição estratificada 80/20 utilizada nos demais cenários, resultando na matriz de confusão apresentada na Figura 20, com as métricas correspondentes consolidadas na Tabela 13.

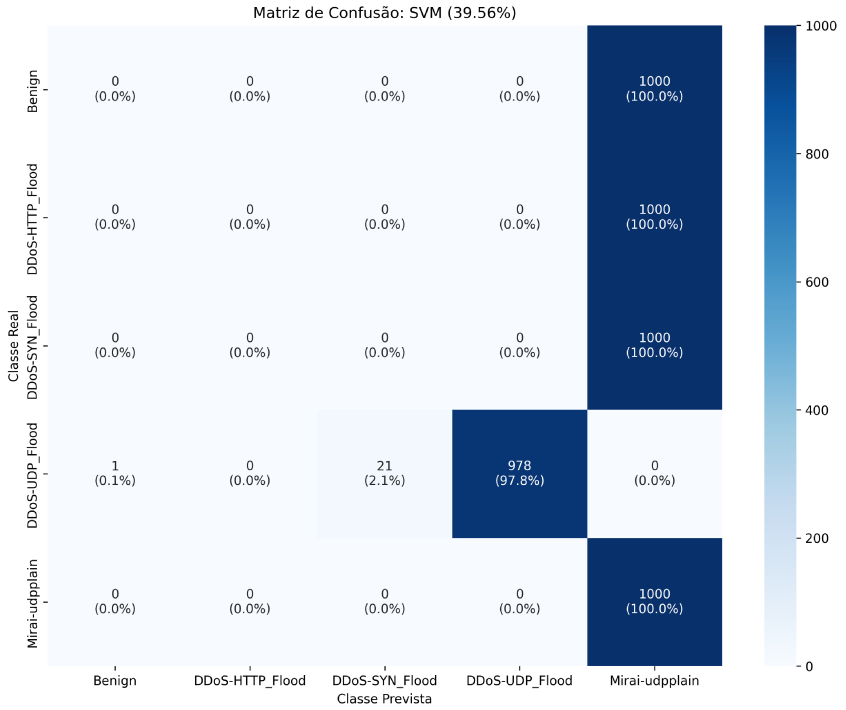

Figura 20 – Top 10 - svm - *SHAP*.

Fonte: elaborado pelo autor.

5.4.2.4 Adaboost

No experimento com o AdaBoost neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 10 do ranqueamento SHAP. O treinamento e a avaliação seguiram a mesma partição estratificada 80/20 utilizada nos demais cenários, resultando na matriz de confusão apresentada na Figura 21, com as métricas correspondentes consolidadas na Tabela 13.

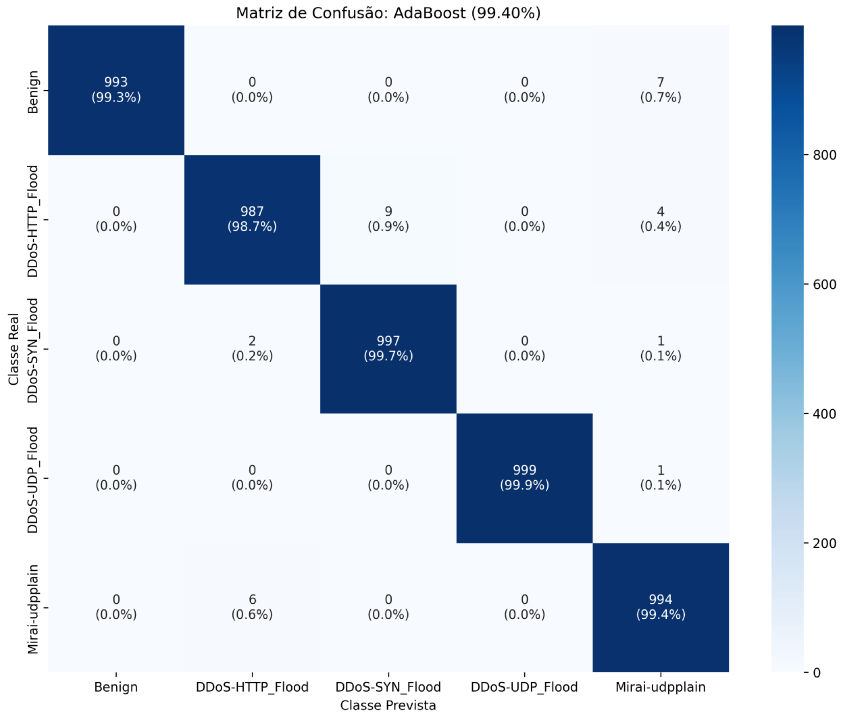

Figura 21 – Top 10 - adaboost - *SHAP*.

Fonte: elaborado pelo autor.

5.4.2.5 MLP

No experimento com a Rede Neural (MLP) neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 10 do ranqueamento SHAP. O treinamento e a avaliação seguiram a mesma partição estratificada 80/20 utilizada nos demais cenários, resultando na matriz de confusão apresentada na Figura 22, com as métricas correspondentes consolidadas na Tabela 13.

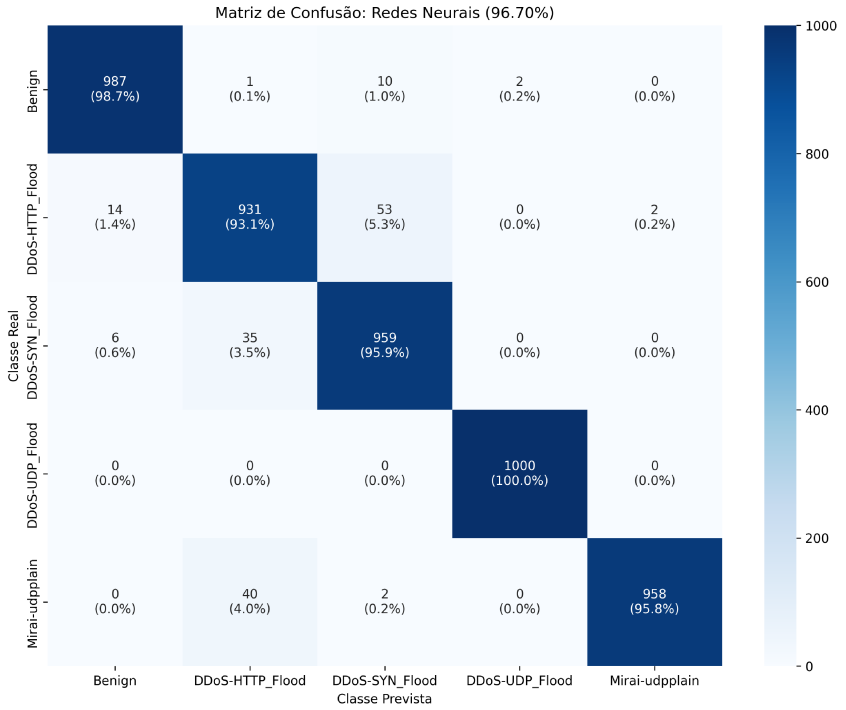

Figura 22 – Top 10 - mlp - *SHAP*.

Fonte: elaborado pelo autor.

5.4.2.6 Resultados Consolidados - Top 10 *features* - SHAP

Tabela 13 – Consolidação das Métricas de Desempenho — Cenário Top 10 Features SHAP.

Modelo	**Acurácia**	**Precisão**	***Recall***	***F1-Score***	***AUC-ROC***
Árvore de Decisão	0.9976	0.9976	0.9976	0.9976	0.9985
Floresta Aleatória	0.9978	0.9978	0.9978	0.9978	0.9998
SVM	0.3956	0.2500	0.3956	0.2778	0.7000
AdaBoost	0.9940	0.9940	0.9940	0.9940	0.9994
Redes Neurais	0.9670	0.9674	0.9670	0.9671	0.9897

Fonte: elaborado pelo autor.

5.4.3 Conjunto - Top 15 *features*

Na terceira e última etapa experimental, os modelos foram submetidos à sua capacidade máxima de dimensionalidade proposta, avaliando o conjunto das 15 características mais relevantes ranqueadas pelo método SHAP. Além das 10 variáveis já consolidadas (*Rate, Header_Length, ack_count, psh_flag_number, Time_To_Live, fin_flag_number, Tot_sum, AVG, syn_flag_number* e *IAT*), foram adicionados *features* de dispersão estatística, protocolo e controle de reinicialização: uso de protocolo seguro (HTTPS), variância dos comprimentos de pacote (*Variance*), valor máximo (Max), desvio padrão (Std) e contagem de *flags de reset* (*rst_flag_number*).

5.4.3.1 Árvore Decisão

No experimento com a Árvore de Decisão neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 15 do ranqueamento SHAP descrito no início desta subseção. O treinamento e a avaliação seguiram a mesma metodologia de divisão estratificada 80/20 empregada nos demais cenários, resultando na matriz de confusão apresentada na Figura 23, com as métricas correspondentes consolidadas na Tabela 14.

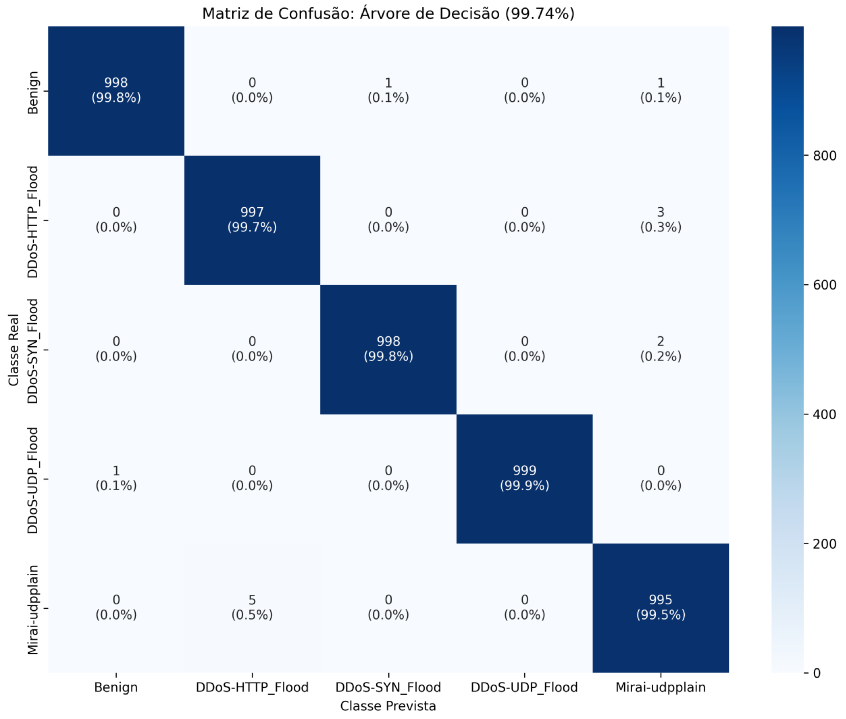

Figura 23 – Top 15 - ad -*SHAP*.

Fonte: elaborado pelo autor.

5.4.3.2 *Random Forest*

No experimento com a Floresta Aleatória neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 15 do ranqueamento SHAP. O treinamento e a avaliação seguiram a mesma partição estratificada 80/20 utilizada nos demais cenários, resultando na matriz de confusão apresentada na Figura 24, com as métricas correspondentes consolidadas na Tabela 14.

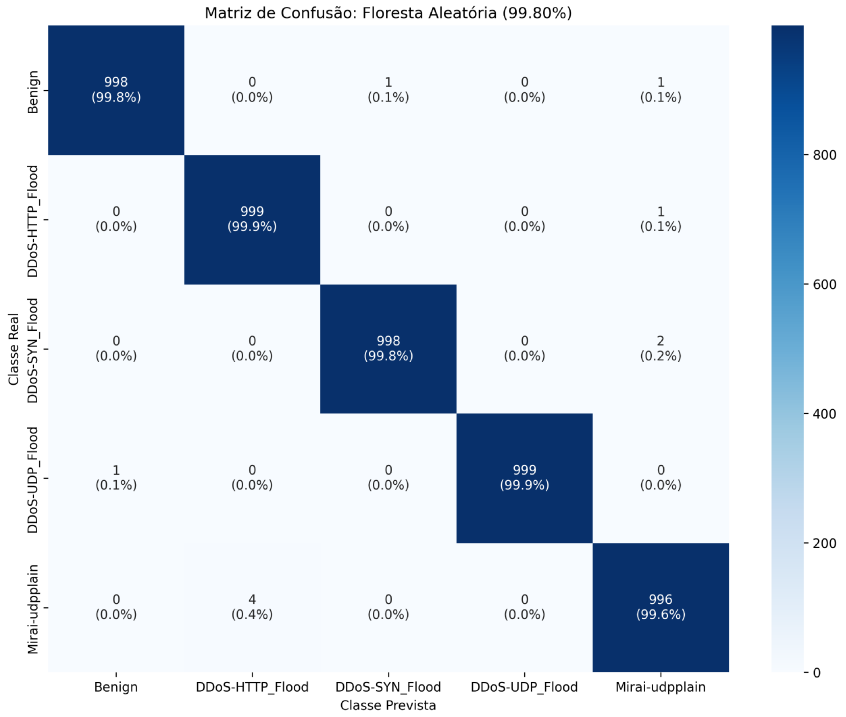

Figura 24 – Top 15 - rf -*SHAP*.

Fonte: elaborado pelo autor.

5.4.3.3 *SVM*

No experimento com a SVM neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 15 do ranqueamento SHAP. O treinamento e a avaliação seguiram a mesma partição estratificada 80/20 utilizada nos demais cenários, resultando na matriz de confusão apresentada na Figura 25, com as métricas correspondentes consolidadas na Tabela 14.

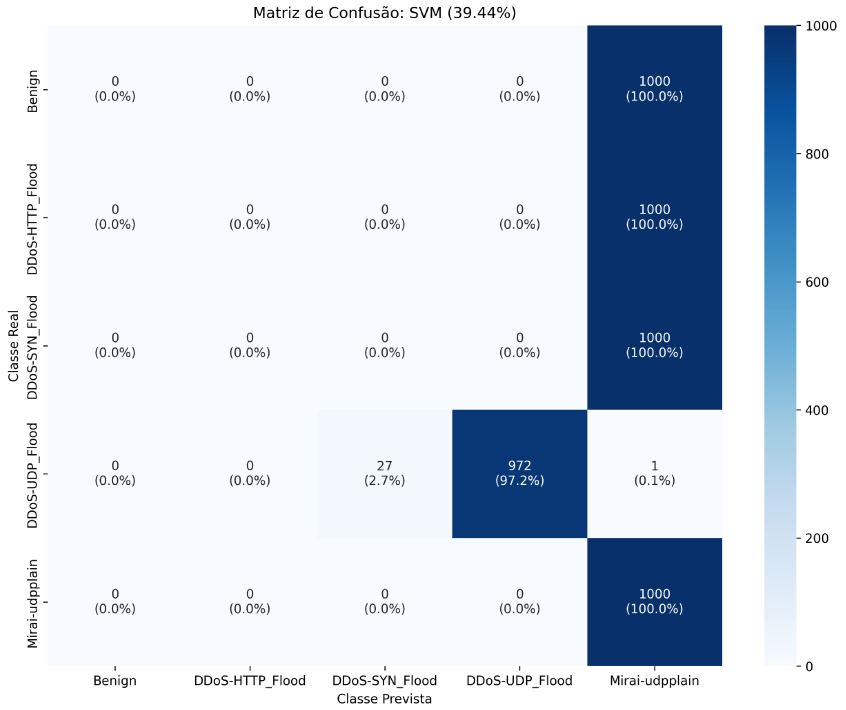

Figura 25 – Top 15 - svm -*SHAP*.

Fonte: elaborado pelo autor.

5.4.3.4 *Adaboost*

No experimento com o *AdaBoost* neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 15 do ranqueamento SHAP. O treinamento e a avaliação seguiram a mesma partição estratificada 80/20 utilizada nos demais cenários, resultando na matriz de confusão apresentada na Figura 26, com as métricas correspondentes consolidadas na Tabela 14.

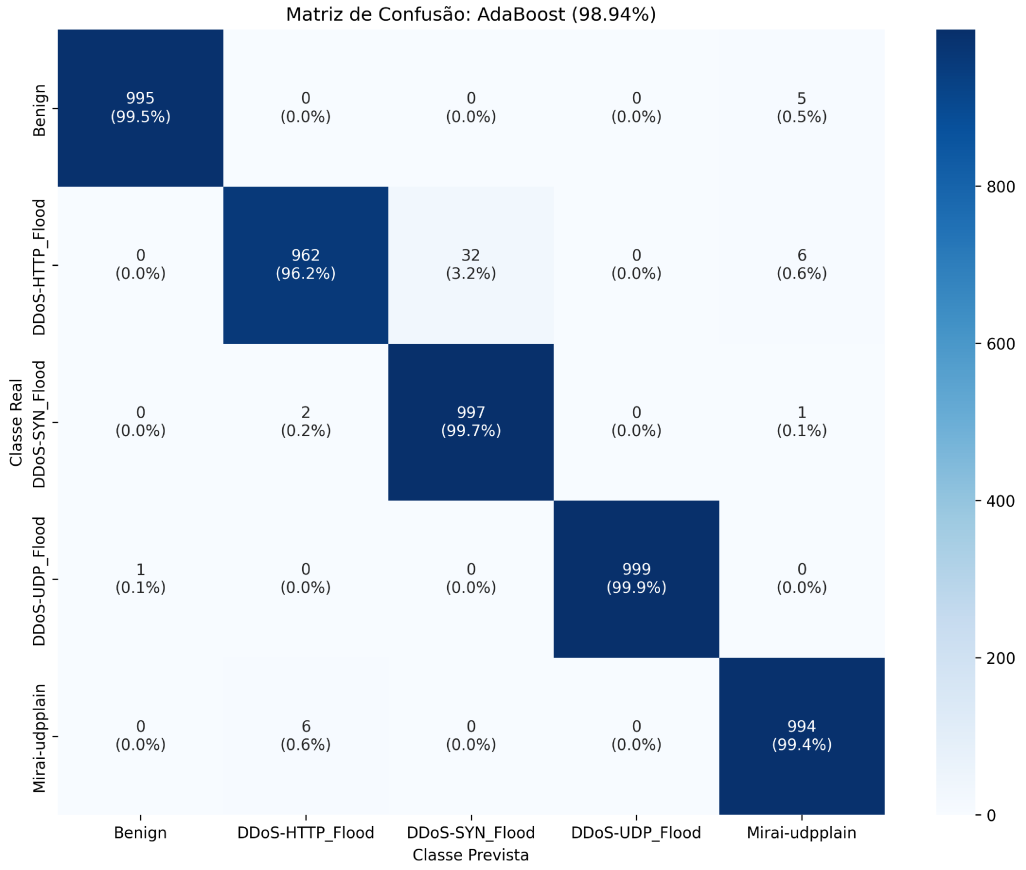

Figura 26 – Top 15 - *Adaboost -SHAP*.

Fonte: elaborado pelo autor.

5.4.3.5 MLP

No experimento com a Rede Neural (MLP) neste cenário, o classificador foi treinado com os mesmos hiperparâmetros apresentados na Tabela 2, restringindo-se o espaço de *features* ao subconjunto Top 15 do ranqueamento SHAP. O treinamento e a avaliação seguiram a mesma partição estratificada 80/20 utilizada nos demais cenários, resultando na matriz de confusão apresentada na Figura 27, com as métricas correspondentes consolidadas na Tabela 14.

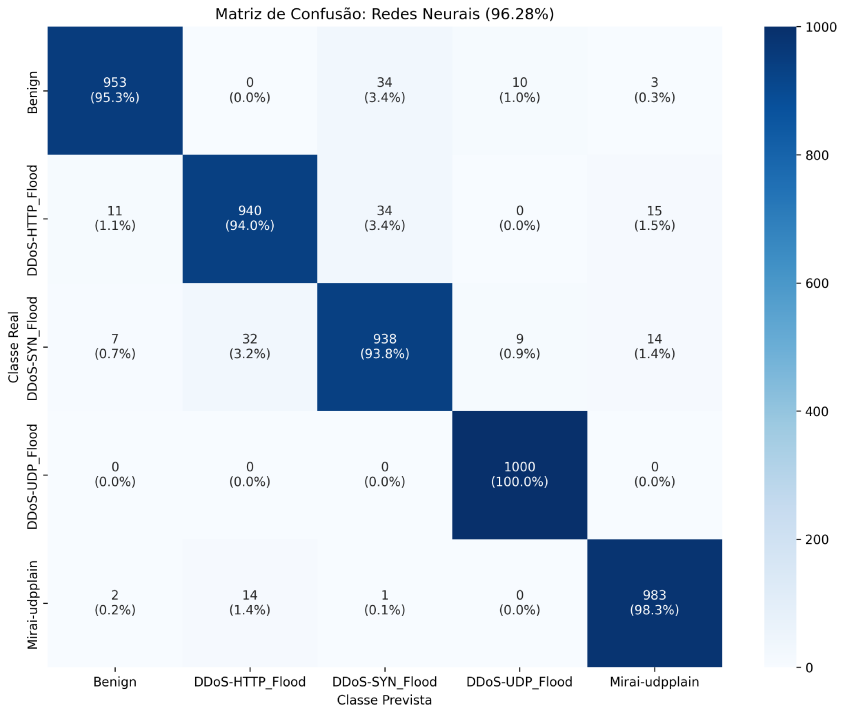

Figura 27 – Top 15 - mlp -*SHAP*.

Fonte: elaborado pelo autor.

5.4.3.6 Resultados Consolidados - Top 15 *features* - SHAP

Tabela 14 – Consolidação das Métricas de Desempenho — Cenário Top 15 *Features SHAP*.

Modelo	**Acurácia**	**Precisão**	***Recall***	***F1-Score***	***AUC-ROC***
Árvore de Decisão	0.9974	0.9974	0.9974	0.9974	0.9984
Floresta Aleatória	0.9980	0.9980	0.9980	0.9980	0.9999
SVM	0.3944	0.2500	0.3944	0.2771	0.7000
AdaBoost	0.9894	0.9895	0.9894	0.9894	0.9995
Redes Neurais	0.9628	0.9628	0.9628	0.9627	0.9869

Fonte: elaborado pelo autor.

6 Análise de Resultados

Este capítulo apresenta uma análise comparativa dos resultados obtidos pelos cinco classificadores nos quatro cenários experimentais considerados: espaço integral de características, denominado *baseline*, e os subconjuntos Top 5, Top 10 e Top 15 definidos por meio do consenso das importâncias SHAP. O objetivo principal é verificar como a redução da dimensionalidade afeta o comportamento preditivo de cada modelo e identificar em quais situações a utilização de um número menor de características preserva ou melhora o desempenho observado no cenário de referência.

6.1 Estratégia de comparação dos cenários

Para isolar o efeito da redução do espaço de entrada, todos os cenários experimentais utilizaram a mesma divisão estratificada de treinamento e teste. Essa decisão metodológica permite realizar uma comparação controlada, pois mantém constantes as instâncias utilizadas na avaliação e altera somente o conjunto de características fornecido aos classificadores. Caso cada cenário fosse avaliado sobre amostras diferentes, uma eventual variação de desempenho poderia decorrer tanto da redução das características quanto de diferenças na dificuldade das instâncias selecionadas.

O conjunto de dados contém 25.000 registros e foi dividido na proporção de 80% para treinamento e 20% para teste. Consequentemente, a partição de teste utilizada nas comparações contém aproximadamente 5.000 registros. Como essa mesma partição foi empregada nos quatro cenários, as diferenças de acurácia podem ser convertidas aproximadamente em quantidades de classificações corretas ou incorretas. Por exemplo, uma diferença de 0,02 ponto percentual corresponde a aproximadamente uma instância do conjunto de teste.

A manutenção da mesma partição de teste é adequada ao objetivo de observar o comportamento dos modelos sob diferentes níveis de redução dimensional. Entretanto, como os resultados foram obtidos a partir de uma única divisão treino-teste, eles representam o comportamento dos classificadores nessa partição específica. Assim, diferenças pequenas devem ser interpretadas como variações pontuais, e não como evidência definitiva de superioridade estatística ou de generalização para outras amostras do conjunto de dados.

A Tabela 15 apresenta a variação da acurácia de cada cenário reduzido em relação ao respectivo *baseline*. Os valores são expressos em pontos percentuais (p.p.). Valores positivos representam aumento da acurácia, enquanto valores negativos representam redução

em relação ao espaço integral de 23 características.

Tabela 15 – Variação da acurácia dos cenários com seleção de características em relação ao *baseline*

Modelo	**Top 5 vs. *baseline***	**Top 10 vs. *baseline***	**Top 15 vs. *baseline***	**Interpretação**
Floresta Aleatória	−0,08 p.p.	−0,02 p.p.	0,00 p.p.	Desempenho praticamente preservado em todos os subconjuntos avaliados.
Árvore de Decisão	−0,06 p.p.	+0,08 p.p.	+0,06 p.p.	Variações pequenas em relação ao *baseline*.
AdaBoost	−3,22 p.p.	+0,44 p.p.	−0,02 p.p.	O Top 5 apresentou perda relevante, enquanto o Top 10 obteve o melhor resultado pontual.
Rede Neural (MLP)	−0,64 p.p.	+4,58 p.p.	+4,16 p.p.	Os subconjuntos Top 10 e Top 15 apresentaram ganhos relevantes nesta execução.
SVM	+0,14 p.p.	+0,14 p.p.	+0,02 p.p.	O desempenho permaneceu baixo, com análise limitada pela ausência de normalização.

Fonte: elaborado pelo autor.
Nota: p.p. significa pontos percentuais. Os valores representam diferenças absolutas de acurácia em relação ao *baseline*, e não variações percentuais relativas.

A comparação demonstra que a redução dimensional não produziu o mesmo efeito em todos os classificadores. Os modelos baseados em árvores mantiveram alto desempenho mesmo com conjuntos de dados reduzidos, enquanto o AdaBoost e a MLP foram mais sensíveis à composição do espaço de entrada. A SVM, por sua vez, manteve baixo desempenho em todos os cenários, indicando que a redução de características, isoladamente, não foi suficiente para corrigir as limitações observadas nessa configuração.

Embora a acurácia permita uma comparação global entre os cenários, sua análise isolada não é suficiente para avaliar um Sistema de Detecção de Intrusão. Neste problema multiclasse, a precisão macro penaliza a atribuição incorreta de instâncias a cada classe, enquanto a revocação macro representa a capacidade média de reconhecer corretamente as cinco categorias avaliadas. O F1-Score sintetiza o equilíbrio entre essas métricas, penalizando desempenhos desiguais. Portanto, mesmo com o conjunto balanceado utilizado neste trabalho, a acurácia deve ser interpretada em conjunto com precisão, revocação e F1-Score.

6.2 Análise por classificador

Tendo sido apresentada, na Seção **??**, a metodologia adotada para comparar os quatro cenários experimentais sobre uma mesma partição de teste, esta seção detalha o comportamento individual de cada um dos cinco classificadores avaliados. Para cada modelo, discute-se a variação da acurácia e das demais métricas multiclasse (precisão, revocação, F1-Score e AUC-ROC) entre o cenário *Baseline* e os subconjuntos Top 5, Top 10 e Top 15 definidos pelo consenso SHAP, buscando identificar padrões de sensibilidade à redução de dimensionalidade específicos de cada arquitetura.

6.2.1 Floresta Aleatória

A Floresta Aleatória apresentou o maior valor absoluto de acurácia entre os classificadores avaliados, alcançando 99,80% tanto no cenário *baseline*, com 23 características, quanto no cenário Top 15. No Top 10, a acurácia foi de 99,78%, o que representa uma diferença de apenas $-0{,}02$ ponto percentual em relação ao espaço integral. No Top 5, o modelo atingiu 99,72%, correspondendo a uma redução de 0,08 ponto percentual.

Considerando que a partição de teste contém aproximadamente 5.000 registros, a diferença entre o Top 10 e o *baseline* corresponde a cerca de uma classificação. Da mesma maneira, a diferença de 0,08 ponto percentual observada no Top 5 corresponde a aproximadamente quatro classificações. Portanto, os resultados indicam que a Floresta Aleatória preservou praticamente todo o seu desempenho após a redução do espaço de entrada.

As demais métricas apresentam comportamento semelhante. O F1-Score variou de 99,72%, no Top 5, a 99,80%, no Top 15 e no *baseline*. A AUC-ROC permaneceu acima de 99,9% nos cenários Top 10, Top 15 e integral, indicando elevada capacidade de separação entre as classes na partição avaliada.

Esse comportamento é compatível com a estrutura da Floresta Aleatória, que agrega as decisões de múltiplas árvores treinadas sobre subconjuntos de amostras e de características. A combinação dos estimadores reduz a sensibilidade do modelo à remoção de atributos individuais. Entretanto, as diferenças entre Top 10, Top 15 e *baseline* são muito pequenas para estabelecer superioridade estatística entre esses cenários.

A principal constatação é que, no experimento realizado, foi possível reduzir o espaço de entrada de 23 para 10 características sem produzir perda relevante de desempenho na Floresta Aleatória.

6.2.2 Árvore de Decisão

A Árvore de Decisão também apresentou comportamento estável. A acurácia do modelo variou de 99,62%, no Top 5, a 99,76%, no Top 10. O *baseline* atingiu 99,68%, enquanto o Top 15 obteve 99,74%.

Em relação ao espaço dimensional, o Top 5 apresentou uma redução de 0,06 ponto percentual. Por outro lado, o Top 10 apresentou aumento de 0,08 ponto percentual e o Top 15 aumento de 0,06 ponto percentual. Considerando o tamanho do conjunto de teste, essas diferenças representam apenas algumas classificações e devem ser interpretadas com cautela.

Apesar de o Top 10 ter apresentado o maior valor pontual, os resultados não permitem afirmar que esse cenário seja estatisticamente superior ao Top 15 ou ao *baseline*. O resultado demonstra, entretanto, que dez características foram suficientes para preservar a capacidade discriminativa da Árvore de Decisão nessa partição experimental.

Assim, a Árvore de Decisão com Top 10 pode ser considerada uma candidata para avaliações futuras em dispositivos de borda, mas sua eficiência operacional ainda deve ser comprovada por meio de medidas de latência, memória e tamanho do modelo.

6.2.3 AdaBoost

O AdaBoost apresentou maior sensibilidade à quantidade e à composição das *features*. No cenário *baseline*, o modelo alcançou acurácia de 98,96%. Com o subconjunto Top 5, a acurácia diminuiu para 95,74%, representando uma perda de 3,22 pontos percentuais. Essa diferença corresponde a aproximadamente 161 classificações no conjunto de teste, sendo consideravelmente maior que as variações observadas na Floresta Aleatória e na Árvore de Decisão.

Quando o espaço de entrada foi ampliado para o Top 10, a acurácia aumentou para 99,40%, superando pontualmente o *baseline* em 0,44 ponto percentual. Esse ganho corresponde a aproximadamente 22 classificações adicionais. No Top 15, a acurácia foi de 98,94%, resultado praticamente idêntico ao espaço integral, com diferença de apenas $-0{,}02$ ponto percentual.

O F1-Score acompanhou esse comportamento, passando de 95,75% no Top 5 para 99,40% no Top 10. A AUC-ROC permaneceu elevada em todos os cenários, variando de 99,68% no Top 5 a 99,95% no Top 15 e no *baseline*. Isso indica que, embora o modelo tenha mantido boa capacidade global de ordenação das classes, o subconjunto Top 5 não foi suficiente para sustentar a mesma qualidade das decisões finais.

6.2.4 Rede Neural MLP

A Rede Neural MLP apresentou o maior ganho de acurácia associado à seleção de características. No cenário *baseline*, o modelo atingiu 92,12%. No Top 5, a acurácia foi de 91,48%, representando redução de 0,64 ponto percentual. No Top 10, entretanto, a acurácia aumentou para 96,70%, correspondendo a um ganho de 4,58 pontos percentuais em relação ao espaço integral. O Top 15 também superou o *baseline*, alcançando 96,28%, com ganho de 4,16 pontos percentuais.

No conjunto de teste utilizado, o aumento de 4,58 pontos percentuais representa aproximadamente 229 classificações corretas adicionais. Diferentemente das pequenas variações observadas nos modelos de árvore, esse ganho possui magnitude relevante no contexto da partição avaliada.

As demais métricas confirmam a melhora. O F1-Score aumentou de 92,06% no *baseline* para 96,71% no Top 10. A AUC-ROC passou de 97,32% para 98,97%, indicando que a redução também melhorou a capacidade do modelo de separar as classes.

Uma interpretação possível é que a retirada de características com menor importância tenha simplificado o espaço de otimização da rede neural. Com menos variáveis de entrada, o modelo pode ter encontrado uma representação mais adequada para as classes avaliadas. No entanto, essa explicação deve ser tratada como hipótese, pois não foram conduzidos experimentos específicos para avaliar a contribuição individual das características removidas.

Também é necessário considerar que a MLP foi treinada sem normalização das variáveis de entrada. Como redes neurais são sensíveis à escala dos atributos, os resultados caracterizam especificamente o comportamento da arquitetura e do pré-processamento utilizados neste trabalho.

6.2.5 SVM

A SVM apresentou acurácia próxima de 39,5% em todos os cenários. O *baseline* atingiu 39,42%, enquanto Top 5 e Top 10 alcançaram 39,56%. No Top 15, a acurácia foi de 39,44%. Embora os cenários reduzidos apresentem variações positivas de até 0,14 ponto percentual, o desempenho permanece baixo em termos absolutos.

A precisão macro foi de 25% em todos os cenários, enquanto o F1-Score permaneceu próximo de 27,7%. Esses resultados indicam que o modelo não apresentou desempenho equilibrado entre as cinco classes. A AUC-ROC de 70%, superior à acurácia e ao F1-Score, sugere que as probabilidades produzidas pelo classificador apresentam algum grau de ordenação entre as classes, mas não resultam em decisões multiclasse adequadas no limiar utilizado.

A redução do número de características não foi suficiente para alterar esse comportamento. Uma explicação provável é a ausência de normalização das variáveis de entrada. O kernel RBF utiliza distâncias no espaço de características e, portanto, é sensível a diferenças de escala. O conjunto de dados combina atributos binários, como as *flags* de protocolo, com variáveis numéricas de magnitude consideravelmente maior, como `Rate`, `IAT` e `Header_Length`. Nessas condições, as variáveis de maior escala podem dominar o cálculo das distâncias.

Dessa forma, os resultados não devem ser interpretados como uma limitação geral da SVM para detecção de ataques DDoS. Eles representam o desempenho da configuração específica avaliada, composta pelo kernel RBF e por dados sem normalização adicional. Uma comparação mais conclusiva exigiria aplicar uma técnica como `StandardScaler` ou `MinMaxScaler`, ajustada exclusivamente sobre o conjunto de treinamento.

6.3 Análise das características selecionadas

O processo de consenso identificou seis características presentes entre as quinze mais relevantes de todos os classificadores: `Rate`, `Header_Length`, `ack_count`, `psh_flag_number`, `Time_To_Live` e `fin_flag_number`. A recorrência dessas características indica que diferentes famílias de modelos utilizaram informações semelhantes para distinguir as cinco classes avaliadas.

A característica `Rate` apresentou o maior valor SHAP médio global, indicando forte participação nas decisões dos classificadores. Esse resultado é coerente com a natureza volumétrica dos ataques DDoS considerados, pois alterações na taxa de transmissão de pacotes podem auxiliar na separação entre tráfego benigno e fluxos de inundação.

A presença recorrente de `Header_Length` indica que o comprimento agregado dos cabeçalhos também contém informação discriminativa no subconjunto analisado. Entretanto, o resultado não permite concluir, isoladamente, que os ataques utilizam cabeçalhos malformados. Os valores SHAP demonstram que o atributo influenciou as decisões dos modelos, mas não estabelecem o mecanismo causal responsável por essa associação.

As variáveis `ack_count`, `psh_flag_number`, `fin_flag_number` e `syn_flag_number` representam informações relacionadas às flags de controle do protocolo TCP. A importância dessas características sugere que diferenças no comportamento das conexões contribuíram para a separação das classes. Essa interpretação é especialmente plausível no caso de ataques SYN Flood, mas sua confirmação exigiria uma análise das distribuições dos atributos em cada classe.

De maneira geral, os resultados SHAP devem ser interpretados como evidências associativas referentes aos padrões aprendidos pelos classificadores. Eles não demonstram

que uma determinada característica seja causa de um ataque nem garantem que o mesmo ranking será obtido em outros dispositivos, períodos de captura ou conjuntos de dados.

6.4 Respostas às questões de pesquisa

Com base na análise consolidada dos resultados apresentados nas seções anteriores, retoma-se, nesta seção, as duas questões de pesquisa formuladas no Capítulo 1 (Seção **??**), respondendo-as à luz das evidências experimentais obtidas. A primeira resposta trata do impacto da redução de dimensionalidade sobre o desempenho dos classificadores, enquanto a segunda discute o equilíbrio entre desempenho preditivo e custo computacional entre os modelos avaliados.

6.4.1 Resposta à RQ1

A primeira questão de pesquisa investiga o impacto dos subconjuntos Top 5, Top 10 e Top 15 no desempenho dos modelos de aprendizado de máquina.

Os resultados demonstram que o efeito da seleção depende do classificador. A Floresta Aleatória e a Árvore de Decisão preservaram praticamente todo o desempenho com conjuntos reduzidos. Para esses modelos, dez características foram suficientes para atingir resultados próximos aos obtidos com as 23 características do *baseline*.

O AdaBoost apresentou perda relevante no Top 5, mas recuperou o desempenho no Top 10. A MLP também apresentou seu melhor resultado com dez características, superando o *baseline* em 4,58 pontos percentuais. A SVM permaneceu com baixo desempenho em todos os cenários, indicando que a redução dimensional não resolveu as limitações da configuração adotada.

Portanto, não foi identificado um único número ótimo de características aplicável a todos os classificadores. No contexto da partição experimental utilizada, o Top 10 apresentou o melhor compromisso preditivo para Árvore de Decisão, AdaBoost e MLP, além de preservar praticamente todo o desempenho da Floresta Aleatória. O Top 5 mostrou-se suficiente para os modelos de árvore, mas produziu perdas mais expressivas para AdaBoost e MLP.

6.4.2 Resposta à RQ2

A segunda questão de pesquisa busca identificar o classificador com melhor equilíbrio entre desempenho preditivo e custo computacional ao utilizar conjuntos reduzidos de *features*.

Em termos exclusivamente preditivos, a Floresta Aleatória apresentou o melhor desempenho global, alcançando 99,80% de acurácia no Top 15 e no espaço integral. No

Top 10, o modelo atingiu 99,78%, preservando praticamente todo o desempenho com redução de aproximadamente 56,5% no número de características de entrada, passando de 23 para 10.

A Árvore de Decisão com Top 10 alcançou acurácia de 99,76%, apenas 0,02 ponto percentual abaixo da Floresta Aleatória no mesmo cenário. Por utilizar uma única estrutura de decisão, esse modelo possui potencial para apresentar menor complexidade de armazenamento e inferência. Entretanto, o trabalho não mediu a profundidade da árvore, o número de nós, o tamanho dos arquivos serializados, a latência de inferência, o consumo de memória ou o consumo energético.

Consequentemente, a RQ2 pode ser respondida apenas parcialmente. A Floresta Aleatória apresentou o melhor desempenho preditivo, enquanto a Árvore de Decisão com Top 10 constitui uma alternativa potencialmente mais simples e com desempenho muito próximo. Não é possível, com as métricas coletadas, afirmar qual dos dois classificadores oferece efetivamente o melhor equilíbrio computacional para dispositivos de borda.

Os resultados permitem apontar ambos como candidatos para uma avaliação futura em hardware real. A determinação do melhor equilíbrio dependerá da medição direta de latência, memória, tamanho do modelo e consumo energético no dispositivo de implantação.

6.5 Limitações da análise

Este trabalho apresenta uma limitação relacionada ao uso de uma única divisão treino-teste. A manutenção da mesma partição foi deliberada e adequada à comparação controlada entre os cenários, pois permitiu alterar somente o número de características. Entretanto, a ausência de múltiplas divisões impede avaliar a estabilidade dos resultados diante de outras composições das amostras. Assim, os valores reportados devem ser interpretados como estimativas pontuais.

Embora não constitua uma limitação, é importante ressaltar que a ausência de normalização das características permitiu manter condições uniformes de pré-processamento entre os classificadores, mas afetou especialmente SVM e MLP, cujos processos de treinamento são sensíveis à escala das variáveis. Consequentemente, a comparação representa o comportamento dos modelos sob o pipeline adotado, e não necessariamente o melhor desempenho que cada algoritmo poderia alcançar.

Por fim, os resultados são específicos ao subconjunto balanceado de cinco classes extraído do CICIoT2023. Em ambientes reais, a distribuição entre tráfego benigno e ataques pode ser significativamente desbalanceada.

7 Conclusão

Este trabalho realizou um estudo comparativo sobre os efeitos da seleção de *features* orientada por consenso SHAP no desempenho de classificadores de aprendizado de máquina aplicados à detecção de ataques de Negação de Serviço Distribuída (DDoS). O interesse em investigar a redução de dimensionalidade decorre das restrições de capacidade computacional, memória e heterogeneidade de protocolos características de ambientes de Internet das Coisas (IoT), que tornam relevante compreender como subconjuntos reduzidos de *features* afetam o desempenho de diferentes famílias de classificadores. Para isso, cinco classificadores de aprendizado de máquina (Árvore de Decisão, Floresta Aleatória, SVM, AdaBoost e Rede Neural) foram treinados e avaliados sobre um subconjunto do *dataset* CICIoT2023, primeiro no espaço integral de *features* e, em seguida, sobre três subconjuntos reduzidos (Top 5, Top 10 e Top 15 *features*), obtidos por meio de um processo de consenso entre os valores de interpretabilidade SHAP calculados individualmente para cada modelo.

Os resultados apresentados no capítulo anterior indicam que os classificadores baseados em árvore (Árvore de Decisão, Floresta Aleatória e AdaBoost) apresentaram o melhor desempenho entre os modelos avaliados, com a Floresta Aleatória atingindo a maior acurácia do estudo (99,80%). O cenário Top 10 *features* mostrou-se um ponto de equilíbrio favorável para o *AdaBoost* e a Rede Neural (MLP), que alcançaram nesse subconjunto seus melhores resultados (99,40% e 96,70%, respectivamente), sem perda relevante de desempenho em relação ao espaço integral de *features*. Já a SVM apresentou acurácia consistentemente baixa (aproximadamente 39,5%) em todos os cenários testados, resultado atribuído à sensibilidade do *kernel* RBF à ausência de normalização das variáveis de entrada, e não a uma falha de convergência do algoritmo, já que o treinamento foi concluído normalmente em todas as execuções.

Vale ressaltar que Os cinco classificadores foram avaliados sob as mesmas condições de pré-processamento, sem etapas adicionais de normalização, refletindo um cenário de baixo custo computacional de preparação dos dados relevantes para o contexto de borda que motiva este trabalho. Sob essa condição uniforme, a SVM apresentou acurácia consistentemente baixa (aproximadamente 39,5%) em todos os cenários testados, evidenciando empiricamente uma característica já bem documentada na literatura: classificadores baseados em margens geométricas, como o *kernel* RBF, são sensíveis à escala das variáveis de entrada, ao contrário dos modelos baseados em árvore, que se mostraram robustos mesmo sem normalização prévia. Esse resultado reforça, na prática, a adequação dos classificadores baseados em árvore para cenários de IoT em que se busca minimizar etapas de pré-processamento no *pipeline* de inferência.

O estudo evidencia que a seleção de *features* orientada por consenso SHAP é capaz de reduzir substancialmente a dimensionalidade do problema sem comprometer e, em parte dos modelos, até melhorando o desempenho dos classificadores avaliados. Esse resultado indica que o consenso entre múltiplos modelos de naturezas distintas pode ser uma estratégia útil para orientar a seleção de *features* em problemas de classificação de tráfego de rede, contribuindo para a discussão sobre como técnicas de interpretabilidade podem ser empregadas não apenas para explicar decisões de modelos já treinados, mas também para apoiar decisões de engenharia sobre quais *features* priorizar. Em particular, observou-se que a Árvore de Decisão, treinada sobre apenas 10 *features*, alcançou desempenho próximo ao dos modelos mais complexos (99,76%), o que ilustra, no contexto estudado, como a redução guiada por consenso SHAP pode preservar a capacidade discriminativa mesmo em modelos estruturalmente simples.

Cabe destacar algumas limitações deste estudo. Primeiro, a avaliação dos modelos foi conduzida sobre uma única divisão treino-teste, sem validação cruzada; os valores de acurácia reportados devem, portanto, ser interpretados como uma estimativa pontual de desempenho, não como uma medida definitiva de generalização. Segundo, os valores de *Shapley* calculados para *SVM*, *AdaBoost* e Rede Neural utilizaram o algoritmo aproximado *KernelExplainer* sobre uma amostra do conjunto de teste, diferente do algoritmo exato empregado para os modelos baseados em árvore, o que introduz uma diferença de precisão entre os *rankings* de importância desses grupos de modelos. Terceiro, os valores SHAP indicam a contribuição de cada *feature* para a decisão dos modelos de natureza associativa , não permitindo inferir relações de causalidade entre os *features* de rede e a ocorrência real dos ataques. Por fim, os resultados são específicos ao subconjunto do *dataset* CICIoT2023 utilizado neste trabalho, de modo que sua validade para outros perfis de tráfego *IoT* ou vetores de ataque não testados ainda não foi estabelecida.

Em conjunto, os resultados obtidos evidenciam o efeito prático que a seleção de *features* via consenso SHAP pode exercer sobre diferentes famílias de classificadores, contribuindo para a compreensão de como essa técnica de interpretabilidade pode ser empregada como ferramenta de apoio à engenharia de *features* em problemas de classificação de tráfego de rede, sujeita às limitações discutidas acima e aos direcionamentos apresentados no capítulo seguinte.

8 Trabalhos Futuros

Como continuidade natural do estudo comparativo desenvolvido, e a partir das limitações apontadas na Conclusão, propõe-se as seguintes direções para trabalhos futuros:

- **Validação em hardware real de borda:** Este trabalho avaliou o desempenho dos classificadores de forma computacional, sem embarcá-los em dispositivos físicos. Como próximo passo, seria necessário transcrever as regras de decisão da Árvore de Decisão e da Floresta Aleatória (treinadas sobre os subconjuntos Top 5 e Top 10) para microcontroladores comerciais (como ESP32, Arduino Portenta ou Raspberry Pi), avaliando latência de inferência, consumo energético e uso de memória em condições reais de operação – métricas que este estudo não mediu diretamente.

- **Reavaliação da SVM e da MLP com normalização de atributos:** Como discutido na Conclusão, os classificadores foram avaliados sob as mesmas condições de pré-processamento, sem normalização adicional. Um desdobramento natural seria repetir os experimentos aplicando técnicas de padronização (como *StandardScaler* ou *MinMaxScaler*) às variáveis de entrada, verificando se o desempenho da SVM e da MLP se aproxima do observado nos modelos baseados em árvore quando essa etapa é incluída.

- **Validação cruzada e testes de significância estatística:** Os resultados deste trabalho foram obtidos a partir de uma única divisão treino-teste. Trabalhos futuros poderiam empregar validação cruzada (*k-fold*) e testes de significância estatística entre os cenários comparados, permitindo estimar intervalos de confiança para os valores de acurácia reportados e fortalecer a robustez das comparações entre modelos e subconjuntos de *features*.

- **Ampliação do escopo de ataques e avaliação de generalização:** Este estudo utilizou um subconjunto do *dataset* CICIoT2023, restrito a cinco classes de tráfego. Seria relevante avaliar se o processo de consenso SHAP produz um ranqueamento de atributos consistente ao ser aplicado a outras variantes de ataque *DDoS*, outras famílias de *botnets* ou outros *datasets* de tráfego *IoT*, de modo a investigar a generalização da abordagem para além do cenário específico analisado.

- **Comparação com outras técnicas de interpretabilidade:** Como este trabalho comparou apenas o efeito da seleção via consenso SHAP, uma extensão natural seria comparar essa abordagem a outros métodos de seleção de atributos e interpretabilidade, como importância por permutação (*permutation importance*) ou LIME,

avaliando se o consenso entre modelos via SHAP oferece vantagens em relação a alternativas computacionalmente mais simples.

- **Padronização da amostragem entre os algoritmos SHAP:** Conforme apontado na Seção 2.6, o cálculo dos valores de Shapley utilizou o *TreeExplainer* (exato) para os modelos baseados em árvore e o *KernelExplainer* (aproximado) para os demais, aplicados sobre amostras de tamanhos distintos. Trabalhos futuros poderiam investigar estratégias para equalizar essa diferença – por exemplo, aumentando a amostra do *KernelExplainer* ou utilizando explicadores alternativos com custo computacional mais homogêneo entre os modelos –, reduzindo a assimetria residual entre os critérios de importância comparados no consenso.

APÊNDICE A – Artefatos do Trabalho

O código-fonte completo desenvolvido neste trabalho encontra-se disponível em repositório público, podendo ser acessado por meio do seguinte endereço eletrônico:

<https://github.com/VitorYuji25/Proejto_TCC_VitorMatsushita_UFU>

O repositório contém todos os artefatos necessários para a reprodução dos experimentos, incluindo scripts, dependências e instruções de execução.

Referências

AL-FUQAHA, A.; GUIZANI, M.; MOHAMMADI, M.; ALEDHARI, M.; AYYASH, M. Internet of things: A survey on enabling technologies, protocols, and applications. **IEEE Communications Surveys & Tutorials**, v. 17, n. 4, p. 2347–2376, 2015. Citado na página 19.

AL-NBHANY, W. A. N. A.; ZAHARY, A. T.; AL-SHARGABI, A. A. **Blockchain-IoT Healthcare Applications and Trends: A Review**. [S.l.]: IEEE, 2024. <https://ieeexplore.ieee.org/document/10379622>. 02 January 2024. Citado 2 vezes nas páginas 19 e 20.

ALOTAIBI, Y.; ILYAS, M. Ensemble-learning framework for intrusion detection to enhance internet of things' devices security. **Sensors**, v. 23, n. 12, p. 5568, 2023. Citado 2 vezes nas páginas 28 e 32.

AWS. **AWS Shield Threat Landscape Report - Q1 2020**. 2020. <https://aws.amazon.com/pt/security/shield/threat-landscape-report/>. Citado na página 21.

AZAM, Z.; ISLAM, M. M.; HUDA, M. N. **Comparative Analysis of Intrusion Detection Systems and Machine Learning-Based Model Analysis Through Decision Tree**. [S.l.]: IEEE, 2023. <https://ieeexplore.ieee.org/document/10185955>. 18 De julho de 2023. Citado na página 24.

BIBI, N.; IQBAL, F.; AKHTAR, S. M.; ANWAR, R.; BIBI, S. A survey of application layer protocols of internet of things. In: **International Journal of Computer Science and Network Security, VOL.21 No.11, November 2021**. [S.l.: s.n.], 2021. p. 1–12. Citado na página 19.

Check Point Research. **Check Point Research reports a 41% increase in global IoT cyber attacks in 2023**. 2023. <https://blog.checkpoint.com/security/check-point-research-reports-a-41-increase-in-global-iot-cyber-attacks-in-2023/>. Acesso em: 10 out. 2024. Citado na página 14.

DONNO, M. D.; DRAGONI, N.; GIARETTA, A.; SPOGNARDI, A. **DDoS-Capable IoT Malwares: Comparative Analysis and Mirai Investigation**. 2018. <https://onlinelibrary.wiley.com/journal/2037>. Citado 2 vezes nas páginas 14 e 21.

FADELE, A. A.; OTHMANA, Z. M.; HASHEM, I. A. T.; ALOTAIBI, F. Internet of Things security: A survey. **Journal of Network and Computer Applications**, v. 88, p. 1–35, 2017. Citado na página 21.

FILHO, H. C. B. A arquitetura rd para seleção de sensores iot baseada em qualidade de contexto no paradigma edge computing. 2020. Citado na página 13.

Fortinet. **What Is DDOS Attack? Learn about DDoS Attack meaning, attack types, and examples.** 2023. <https://www.fortinet.com/resources/cyberglossary/ddos-attack>. Citado 3 vezes nas páginas 21, 22 e 23.

_____. **What is Network Traffic?** 2023. <https://www.fortinet.com/resources/cyberglossary/network-traffic>. Citado 2 vezes nas páginas 22 e 23.

GOYAL, S. B.; THAKUR, N.; QADEER, S. A. Bridging the interpretability gap: A shap-enhanced framework for intrusion detection in cybersecurity. **International Journal of Management and Data Intelligence**, v. 1, n. 1, p. 1–13, 2025. ISSN 3139-4922. Citado 2 vezes nas páginas 29 e 32.

IBM. **O que é um ataque distributed denial-of-service (DDoS)?** 2022. <https://www.ibm.com/br-pt/topics/ddos>. Citado na página 23.

KUCHUK, H.; MALOKHVII, E. Integration of iot with cloud, fog, and edge computing: a review. **Advanced Information Systems**, v. 8, n. 2, p. 65–78, 2024. Citado na página 13.

LE, T.-T.-H.; WARDHANI, R. W.; PUTRANTO, D. S. C.; JO, U.; KIM, H. Toward enhanced attack detection and explanation in intrusion detection system-based iot environment data. **IEEE Access**, v. 11, p. 131661–131676, 2023. Citado 2 vezes nas páginas 27 e 32.

LUNDBERG, S. M.; LEE, S.-I. A unified approach to interpreting model predictions. In: GUYON, I.; LUXBURG, U. V.; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems**. Long Beach, CA, USA: Curran Associates, Inc., 2017. v. 30, p. 4765–4774. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf>. Citado na página 26.

MAGáN-CARRIóN DANIEL URDA, I. D.-C. R.; DORRONSORO, B. Towards a reliable comparison and evaluation of network intrusion detection systems based on machine learning approaches. **Appl. Sci. 2020**, 2020. Citado na página 24.

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**, MDPI, v. 23, n. 13, p. 5941, 2023. Citado na página 15.

OLIVEIRA, F. B. d. Framework para detecção de ataques dos em dispositivos iot, utilizando abordagens de aprendizado de máquinas. 2023. Citado na página 14.

Oracle. **What is IoT?** 2020. <https://www.oracle.com/internet-of-things/>. Acesso em: 10 out. 2024. Citado na página 13.

OSTEC. **IDS: História, Conceito e Terminologia**. 2018. <https://ostec.blog/en/perimeter/ids-history-concept-terminology/>. Acesso em: 1 mar. 2018. Citado na página 24.

Palo Alto Networks. **O que é um Ataque Distribuído de Negação de Serviço (DDoS)?** 2024. <https://www.paloaltonetworks.com/cyberpedia/what-is-a-ddos-attack>. Citado 3 vezes nas páginas 13, 14 e 21.

SANTOS, B. P.; SILVA, L. A. M.; CELES, C.; BORGES, J.; PERES, B.; VIEIRA, M. A. M.; VIEIRA, L. F. M.; GOUSSEVSKAIA, O.; LOUREIRO, A. A. F. **Internet das Coisas da teoria à prática**. 2023. <https://homepages.dcc.ufmg.br/~mmvieira/cc/papers/internet-das-coisas.pdf>. Citado na página 18.

SONAR, K.; UPADHYAY, H. **A Survey: DDOS Attack on Internet of Things**. 2014. <http://www.ijerd.com/paper/vol10-issue11/Version_3/I10115863.pdf>. Citado 2 vezes nas páginas 13 e 19.

VERMA, P.; DUMKA, A.; SINGH, R.; ASHOK, A.; GEHLOT, A.; MALIK, P. K.; GABA, G. S.; HEDABOU, M. A novel intrusion detection approach using machine learning ensemble for iot environments. **Applied Sciences**, v. 11, n. 21, p. 10268, 2021. Citado 2 vezes nas páginas 31 e 32.

ZHANG, H.; UPADHYAY, D.; ZAMAN, M.; SAMPALLI, S. Lightweight iot intrusion detection via feature and sample reduction with multi-client ensemble for zero-day attacks. **IEEE Access**, v. 14, p. 29764–29780, 2026. Citado 3 vezes nas páginas 30, 31 e 32.