UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM REDES DE COMPUTADORES

DAVI BEZERRA YADA DA SILVA

APLICAÇÃO DE PRIVACIDADE DIFERENCIAL NA DETECÇÃO DE ATAQUES EM INTERNET DAS COISAS

QUIXADÁ

2025

DAVI BEZERRA YADA DA SILVA

APLICAÇÃO DE PRIVACIDADE DIFERENCIAL NA DETECÇÃO DE ATAQUES EM INTERNET DAS COISAS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Redes de Computadores do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de Tecnólogo em Redes de Computadores.

Orientador: Prof. Dr. Jeandro de Mesquita Bezerra.

QUIXADÁ

2025

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

S579a Silva, Davi Bezerra Yada da.
Aplicação de privacidade diferencial na detecção de ataques em internet das coisas / Davi Bezerra Yada da Silva. – 2025.
57 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Redes de Computadores, Quixadá, 2025.
Orientação: Prof. Dr. Jeandro de Mesquita Bezerra.

1. Privacidade Diferencial. 2. Internet das Coisas. 3. Detecção de Ataques. 4. TinyML. I. Título.
CDD 004.6

DAVI BEZERRA YADA DA SILVA

APLICAÇÃO DE PRIVACIDADE DIFERENCIAL NA DETECÇÃO DE ATAQUES EM INTERNET DAS COISAS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Redes de Computadores do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de Tecnólogo em Redes de Computadores.

Aprovada em: 12/03/2025.

BANCA EXAMINADORA

Prof. Dr. Jeandro de Mesquita Bezerra (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Francisco Helder Candido dos Santos Filho
Universidade Federal do Ceará (UFC)

Prof. Dr. Roberto Cabral Rabêlo Filho
Universidade Federal do Ceará (UFC)

À minha mãe e meu pai, que sempre estiveram comigo. Não teria sido o mesmo sem vocês. Este trabalho é resultado de um esforço conjunto, pois vocês fazem parte dessa trajetória.

AGRADECIMENTOS

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

Ao Prof. Dr. Jeandro de Mesquita Bezerra, pela paciência e dedicação ao avanço do projeto, além das oportunidades de aprendizado que me ofereceu.

Aos professores participantes da banca examinadora Francisco Helder Candido dos Santos Filho e Roberto Cabral Rabêlo Filho pelo tempo, pelas valiosas colaborações e sugestões.

Um agradecimento especial à minha família, que sempre ofereceu apoio incondicional durante minha jornada acadêmica. Cada um de vocês tem meu mais sincero agradecimento.

Ao meu querido pai e minha querida mãe, por me darem o conforto e suporte necessários. Às minhas queridas avós, pela dedicação e amor. Aos meus irmãos Fátima Bezerra e Rafael Bezerra, por tornarem os momentos divertidos e alegres. Nossa amizade é de grande importância.

Um agradecimento aos meus grandes amigos Eduardo Katsuo e Pedro Yuzuki, que acompanharam de perto o andamento do trabalho. As horas de estudo e diversão com vocês serão sempre lembradas. Dedico a vocês a música "With a Little Help From my Friends", da banda *The Beatles*.

Ao meu grande amigo Luciano "Pivas", por todo o apoio e amizade.

Ao meu amigo Gabriel Alves, pelos conselhos e amizade, além das incontáveis horas de discussão e aprendizado mútuo.

Aos meus colegas de curso e amigos que fiz durante a jornada acadêmica, com quem convivi durante os últimos anos, pela troca de experiências e companheirismo.

A todos que participaram, direta ou indiretamente, do desenvolvimento deste trabalho de pesquisa, enriquecendo o meu processo de aprendizado.

RESUMO

A crescente adoção da Internet das Coisas amplia desafios de segurança, especialmente no uso de aprendizado profundo para detecção de ataques, uma vez que esses modelos estão suscetíveis a vazamentos de dados sensíveis. Este trabalho investiga a aplicação de Privacidade Diferencial como defesa contra ataques de inferência de associação, que exploram modelos treinados para recuperar informações dos dados utilizados no aprendizado. Para isso, foi implementado um classificador neural diferencialmente privado utilizando *tinyml*, permitindo sua execução em dispositivos de borda com recursos limitados. Os experimentos mostraram que, enquanto o modelo *baseline* obteve 99% de acurácia, a aplicação de PD reduziu esse valor para até 64%, dependendo do nível de ruído adicionado. No entanto, a privacidade diferencial diminuiu significativamente a eficácia do ataque de inferência, reduzindo sua acurácia de 70% para 52% nos cenários mais restritivos. Além disso, a otimização via *tinyml* permitiu uma execução até 10 vezes mais rápida, reduzindo o consumo de memória de 8,35 MB para apenas 0,05 MB. Esses achados reforçam a viabilidade da privacidade diferencial como estratégia para mitigar riscos de vazamento em modelos de aprendizado profundo aplicados à segurança de redes de Internet das Coisas, destacando um importante *trade-off* entre privacidade e desempenho.

Palavras-chave: privacidade diferencial; internet das coisas; detecção de ataques; tinyml.

ABSTRACT

The increasing adoption of the Internet of Things expands security challenges, especially in the use of deep learning for attack detection, as these models are susceptible to sensitive data leaks. This work investigates the application of Differential Privacy as a defense against membership inference attacks, which exploit trained models to recover information from the data used in learning. For this purpose, a differentially private neural classifier was implemented using tinyml, allowing its execution on edge devices with limited resources. The experiments showed that while the baseline model achieved 99% accuracy, the application of DP reduced this value to up to 64%, depending on the level of noise added. However, differential privacy significantly decreased the effectiveness of the inference attack, reducing its accuracy from 70% to 52% in the most restrictive scenarios. Furthermore, tinyml optimization enabled execution up to 10 times faster, reducing memory consumption from 8.35 MB to just 0.05 MB. These findings reinforce the feasibility of differential privacy as a strategy to mitigate leakage risks in deep learning models applied to IoT network security, highlighting an important trade-off between privacy and performance.

Keywords: differential privacy; internet of things; attack detectin; tinyml.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AF	Aprendizado Federado
AM	Aprendizado de Máquina
AP	Aprendizado Profundo
CB	Computação de Borda
CN	Computação de Nuvem
DP-ADAM	Estimativa de Momento Adaptativa Diferencialmente Privada
DP-SGD	Descida Gradiente Estocástico Diferencialmente Privada
FN	*Falsos Negativos*
FP	*Falsos Positivos*
IoT	Internet das coisas, do inglês "Internet of Things"
IP	*Internet Protocol*
LSTM	*Long Short-Term Memory*
MIA	Ataque de Inferência de Associação, do inglês "Membership Inference Attack"
MLP	*Multilayer Perceptron*
PATE	Agregação Privada de Conjunto de Professores
PD	Privacidade Diferencial
SDI	Sistemas de Detecção de Intrusão
SGD	Descida Gradiente Estocástico
SMOTE	*Synthetic Minority Over-sampling Technique*
SSH	*Secure Shell*
Ts	*Timestamp*
VN	*Verdadeiros Negativos*
VP	*Verdadeiros Positivos*

LISTA DE SÍMBOLOS

$\mathscr{A}$	Algoritmo de tratamento de dados
D_i	Dataset
$\in$	Pertence a
$\leq$	Menor ou igual a
Pr	Probabilidade de
S	Subconjunto
ε	Parâmetro de privacidade diferencial, epsilon
∞	Infinito
c	Limite de corte
B	Tamanho de Micro-Batch
α	Multiplicador de Ruído

SUMÁRIO

1 INTRODUÇÃO

O crescente uso da Internet das coisas, do inglês "Internet of Things" (IoT) possibilitou um grande avanço em diversos domínios no mundo atual, como monitoramento ambiental, automação residencial e predial e redes inteligentes, com dispositivos embarcados e objetos domésticos conectados à internet trocando dados entre sistemas para aplicações úteis no cotidiano (Mukhopadhyay; Suryadevara, 2014). Essas aplicações estão cada vez mais presentes na vida dos usuários, tendo destaque no contexto de casas e cidades inteligentes, saúde e indústrias (Wang *et al.*, 2020). Com o avanço exponencial da IoT, surge a necessidade de processar dados de forma mais eficiente e rápida, o que impulsiona os paradigmas de Computação de Borda (CB) e *TinyML*. A CB reduz custos de latência, processamento e tráfego (Cao *et al.*, 2020), enquanto o *tinyml* permite a execução de modelos de Aprendizado de Máquina (AM) e Aprendizado Profundo (AP) em dispositivos limitados para aplicações IoT (Abadade *et al.*, 2023), o que diminui a necessidade de grandes infraestruturas. No entanto, ao usar essas tecnologias, devemos tomar medidas de cibersegurança para proteger os dispositivos e dados dos usuários, devido à sua natureza de restrição de recursos e vulnerabilidades (Rodrigues; Abreu, 2021).

Ferramentas de cibersegurança são essenciais para a proteção do tráfego IoT, como criptografia e protocolos de segurança. Porém, os dispositivos ainda são alvos de ciberameaças que exploram as vulnerabilidades dos dispositivos e a privacidade dos dados de usuários, como ataques de análise de tráfego (Alfandi *et al.*, 2021). Apesar do uso de criptografia manter o sigilo dos dados, estudos anteriores mostram que observadores podem usar dados de rede criptografados e informações sobre pacotes, como tamanho e tempo de chegada, para identificar informações de usuários IoT [Pinheiro et al. 2021]. Recentemente, modelos de aprendizado de máquina e aprendizado profundo têm utilizado esses dados para o treinamento devido ao seu potencial de revelar padrões complexos e construir sistemas inteligentes (Tsimenidis *et al.*, 2022). Entretanto, o uso desses dispositivos levanta preocupações com a privacidade, pois os dados gerados podem ser sensíveis e vulneráveis a vazamentos ou exploração indevida por meio de modelos de aprendizado supervisionado. Apesar da capacidade de melhorar ferramentas de proteção para IoT, os modelos de aprendizado profundo são alvos de ataques à privacidade, como MIA, e podem expor os dados empregados durante o treinamento (Boulemtafes *et al.*, 2020). Esses ataques identificam os dados usados no treinamento do modelo, comprometendo informações sensíveis como localizações frequentes, horários de uso e preferências pessoais.

Dadas as vulnerabilidades citadas, existem trabalhos que exploram diferentes abor-

dagens para preservação de privacidade. (Pedro, 2022) propôs o uso de Privacidade Diferencial Local, que ofusca os dados localmente antes de serem transferidos e armazenados, garantindo maior proteção durante a transmissão. Outra linha de pesquisa envolve o uso de Aprendizado Federado com *tinyml* e Computação de Borda (Ren *et al.*, 2023), permitindo que os dispositivos treinem modelos localmente sem compartilhar diretamente os dados brutos. Estudos também exploraram o uso de Privacidade Diferencial (PD), um modelo matemático para anonimização de dados proposto por (Dwork, 2006), no treinamento de modelos. O algoritmo Descida Gradiente Estocástico Diferencialmente Privada (DP-SGD), utilizado no trabalho de (Pustozerova *et al.*, 2023), adiciona ruído ao processo de treinamento para proteger os dados utilizados em redes neurais profundas e aprendizado federado.

Quanto às ameaças a sistemas IoT, trabalhos anteriores propuseram abordagens de Sistemas de Detecção de Intrusão (SDI) para prevenir e mitigar ataques de rede, como demonstrado por (Jyothsna *et al.*, 2011). Recentemente, SDIs têm sido combinados com técnicas de aprendizado de máquina para melhorar suas funcionalidades. Essas soluções deixam em aberto a possibilidade de integrar PD às tecnologias propostas, focando na ofuscação direta de dados para aumentar a privacidade.

Em relação aos pontos em abertos das soluções citadas, este trabalho propõe o uso de privacidade diferencial para desenvolvimento de um modelo de aprendizado profundo baseado em *TinyML*. Essa abordagem visa mitigar vazamentos de dados por modelos de aprendizado profundo contra ataques adversariais, como o MIA, em funcionamento na borda de rede IoT. Desenvolvemos um modelo de aprendizado profundo do tipo *FeedForward* diferencialmente privado. Implementou-se esse modelo com *tinyML* para classificar de rede para IoT de forma otimizada em baixo custo.

1.1 Objetivo Geral

O Objetivo Geral deste trabalho é implementar um sistema de detecção de ataques diferencialmente privado em ambiente IoT.

1.2 Objetivos Específicos

Os objetivos específicos incluem:

1. Implementar um modelo *FeedForward* baseado em *tinyML* para borda IoT;

2. Ofuscar o processo de treinamento do modelo com privacidade diferencial e;
3. Diminuir o desempenho de ataques de inferência de associação com DP-SGD;

1.3 Organização

O restante do trabalho está organizado da seguinte forma: O Capítulo 2 apresenta trabalhos relacionados à PD, *tinyml* e detecção de intrusão. O Capítulo 3 descreve a fundamentação teórica de alguns conceitos abordados ao longo do trabalho. O Capítulo 4 demonstra a metodologia e explica as etapas de desenvolvimento do projeto. Enfim, o Capítulo 5 reúne os resultados extraídos de experimentos do projeto.

2 TRABALHOS RELACIONADOS

Este capítulo traz um resumo dos trabalhos relacionados com abordagens de Privacidade Diferencial e Detecção de anomalias. Apresentamos as principais propostas, metodologias e contribuições dos trabalhos relacionados para o avanço da área.

O trabalho de (Machooka *et al.*, 2025) implementa duas abordagens de privacidade diferencial para detecção de intrusão em IoT, o DP-SGD e Agregação Privada de Conjunto de Professores (PATE). Os autores destacam a privacidade diferencial em um modelo *Long Short-Term Memory* (LSTM) e um modelo *Multilayer Perceptron* (MLP) para detecção de intrusão, realizando uma comparação entre as duas abordagens de privacidade. A avaliação é feita comparando os modelos obtidos com os diferentes níveis de privacidade em cada abordagem.

Os autores de (Villegas-Ch *et al.*, 2024) abordam a construção de um sistema de aprendizado federado baseado em *tinyml* para uma rede IoT com privacidade de dados. O sistema envolve a coleta e processamento de dados através de dispositivos sensores que captam e usam os dados para iterações de treinamento do modelo de Aprendizado Federado (AF). Para garantir segurança e privacidade, os autores mencionam o uso de criptografia no envio de atualizações e citam estratégias para privacidade diferencial, sem detalhar a técnica específica utilizada. Os autores avaliam o consumo energético e a precisão do modelo, tendo uma queda de 33% e 1,2%, respectivamente.

Em (Pustozerova *et al.*, 2023), avaliou-se o *trade-off* entre utilidade e privacidade em um modelo de aprendizado federado, utilizando os conjuntos de dados *Purchase-100* e *LendingClub-Loan*. A privacidade é implementada através de DP-SGD e saída ofuscada de inferência. Os autores utilizam um ataque de inferência de associação do tipo caixa-preta baseado em modelos sombra para avaliar a privacidade alcançada. Os experimentos são definidos com várias configurações, estabelecendo 5 níveis de nós clientes de treinamento (de 2 a 32) e diferentes parâmetros de privacidade. Os resultados demonstram que o DP-SGD mantém um melhor *trade-off* e diminui o desempenho dos ataques de inferência.

A Tabela 1 mostra um breve resumo do que os autores abordaram em seus trabalhos.

Comparando os trabalhos, observa-se que os estudos de (Machooka *et al.*, 2025) e (Pustozerova *et al.*, 2023) focam diretamente em privacidade diferencial aplicada ao treinamento de modelos, enquanto o trabalho de (Villegas-Ch *et al.*, 2024) trata de aprendizado federado, mas sem detalhar as técnicas específicas de privacidade diferencial. Além disso, o trabalho de (Machooka *et al.*, 2025) explora o impacto de diferentes modelos e níveis de privacidade sobre as

Tabela 1 – Comparação de trabalhos relacionados

Autor	**Tecnologias**	**Técnica de privacidade**	**Avaliação**	**Métricas**
(Machooka *et al.*, 2025)	LSTM, MLP	DP-SGD, PATE	Comparação	Precisão, acurácia, recall, F1-score
(Villegas-Ch *et al.*, 2024)	AF	Não citado	Desempenho e utilidade na rede	Precisão e consumo energético
(Pustozerova *et al.*, 2023)	Aprendizado Federado	DP-SGD, saída ofuscada	Defesa contra MIA	Métricas de ataque, precisão, acurácia, recall, F1-score
Proposto. 2025	*FeedForward*, *Tinyml*	DP-SGD	Defesa contra MIA	Métricas de ataque, precisão, acurácia, recall, F1-score

Fonte: Elaborado pelo autor (2025).

métricas de desempenho, enquanto (Pustozerova *et al.*, 2023) enfatiza a defesa contra ataques de inferência. Já o trabalho de (Villegas-Ch *et al.*, 2024) destaca o impacto em termos de consumo energético e precisão, abordando os desafios práticos de implementação em redes IoT. Nosso trabalho utiliza abordagens semelhantes de privacidade diferencial, com foco em um modelo de detecção de ataques otimizado para a borda de rede IoT, tendo como avaliação a defesa contra ataques MIA.

3 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem por objetivo apresentar os principais conceitos abordados neste trabalho. A Seção 3.1 apresenta sobre Internet das Coisas (IoT), tendo em seguida detalhes sobre a Computação de Borda. A Seção 3.2 detalha sobre o Aprendizado Profundo, mostrando suas características e utilidades, além de fornecer uma explicação sobre a classificação de ataques e apresentar a tecnologia *tinyml*. Por fim, a Seção 3.3 traz um resumo sobre a privacidade diferencial, tendo em seguida a Seção 3.4 com informações sobre ataques de inferência de associação MIA.

3.1 Internet das coisas e Computação de Borda

3.1.1 Internet das Coisas

A internet das coisas consiste em dispositivos conectados entre si através de equipamentos e protocolos, com objetos equipados com sensores, atuadores e *softwares* que realizam coleta e análise de grandes volumes de dados. O surgimento da IoT com suas características nos trouxe diversas oportunidades, desenvolveu setores de mercado e apresentou questões de pesquisa e possibilidades de tecnologias e aplicações. Grande parte dos dispositivos IoT possui restrição de recursos, como baixa capacidade de processamento, memória limitada, baixo armazenamento e menor consumo de energia (Gyamfi; Jurcut, 2022).

A capacidade dos dispositivos de obter dados sobre ambientes, sistemas e outras aplicações permite melhor tomada de decisão, automação de tarefas, otimização de processos e possibilidade de implementação em diversos domínios, como saúde, indústria, transporte, casas inteligentes e cidades inteligentes. Esses dispositivos são normalmente conectados à estruturas de rede através de meios de comunicação sem fio, como tecnologias 4G/5G, *Bluetooth, ZigBee e Wi-fi*, ou cabeadas, como *Ethernet* (Carnaz; Nogueira, 2016). Os dispositivos com chip *Wi-fi* aumentam a conectividade, tornando-se mais acessíveis e mais amplamente adotados devido ao seu funcionamento baseado em redes TCP/IP (Chen *et al.*, 2020). Com a constante proliferação de dispositivos e aplicações IoT, a necessidade de manter redes cada vez mais estáveis e níveis de consumo de energia e latência torna-se vital.

3.1.2 Computação de Borda

A Computação de Borda (CB) é um paradigma computacional para processamento de dados mais próximo às fontes de dados, que surge como uma alternativa à Computação de Nuvem (CN). A CN, apesar de prover recursos computacionais abundantes, em determinados contextos não é ideal para ambientes em que a largura de banda, latência e consumo de energia devem ser mínimos, como ambientes industriais de risco. Nesse sentido, a CB se porta mais perto dos usuários e fontes de dados, trazendo serviços mais rápidos e processamento em baixa escala (Cao *et al.*, 2020). Assim, a CB traz o processamento para perto dos dispositivos e permite a atuação local, sem a necessidade de longas transferências de dados e uso de largura de banda, comparado com a computação em nuvem. Arquiteturas de computação de borda devem considerar a segurança como prioridade, pois os dispositivos estão vulneráveis a intrusões e outros tipos de ataque, além de lidarem com dados sensíveis (Khan *et al.*, 2019).

Figura 1 – Exemplo de uso da Computação de Borda

Fonte: Elaborada pelo autor.

No contexto de internet das coisas, a computação de borda pode ser utilizada para aproveitar melhor as capacidades de processamento e dar respostas mais rápidas aos dispositivos, evitando alta latência devido a fatores de conexão e servidores e riscos na transferência de dados sensíveis. A Figura 1 ilustra uma arquitetura exemplificada em (Shi; Dustdar, 2016), em que os dispositivos são usados para procurar uma criança desaparecida. Nesse cenário, câmeras

de segurança e de outros dispositivos atuam como dispositivos finais que capturam a imagem de uma criança desaparecida. Se essa atividade utilizasse computação em nuvem, o custo de transferência e a privacidade seriam críticos, por conta da natureza desse paradigma, além da demora para resolução da ocorrência. Com a CB, os dispositivos enviariam as informações para os nós de borda, que atuariam com melhor processamento para pesquisar as imagens e apenas relatariam à nuvem suas descobertas sobre a criança desaparecida.

3.2 Aprendizado Profundo, Classificação de Ataques e TinyML

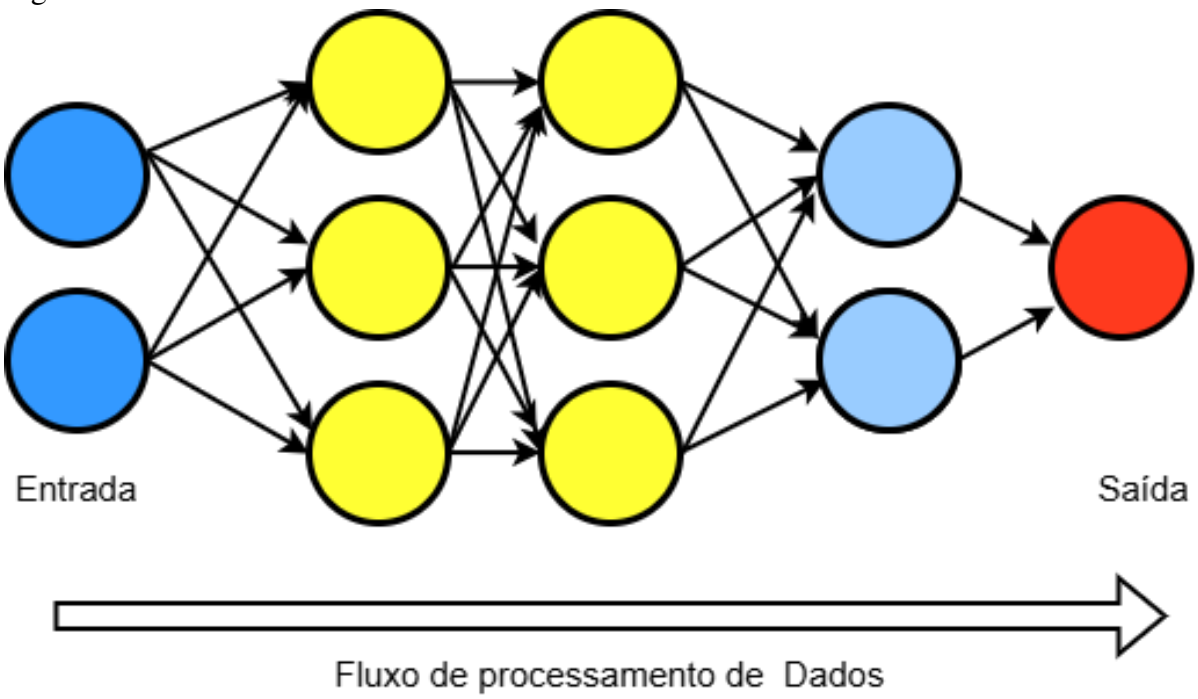
3.2.1 Aprendizado Profundo

O Aprendizado Profundo (AP) é uma subárea do aprendizado de máquina que utiliza diversas camadas de processamento interligadas para representar e aprender abstrações de dados e construir modelos computacionais. Utilizando um conjunto de dados, cada camada computacional tende a alterar seus parâmetros de aprendizado a partir da representação da camada anterior utilizando retropropagação (Pouyanfar *et al.*, 2018). Essa estrutura de processamento de dados é capaz de criar modelos computacionais para atividades de visão computacional, reconhecimento de padrões em dados, reconhecimento de voz e imagem, classificação e outros domínios (LeCun *et al.*, 2015).

Um tipo de modelo de aprendizado profundo é a rede neural profunda *feedforward*, um modelo computacional composto de múltiplas camadas de processamento totalmente interconectadas, capaz de representar e aprender abstrações de dados de forma progressiva (Reis, 2021). Cada camada recebe a saída da anterior, ajusta seus parâmetros para otimizar o aprendizado sobre os dados e, em seguida, passa uma nova abstração para a camada seguinte (Pouyanfar *et al.*, 2018). A arquitetura *feedforward* descreve o fluxo das conexões de aprendizado, indo da entrada até a saída, passando pelas camadas ocultas (Bezerra, 2016). A Figura 2 demonstra sua estrutura básica. Utilizou-se um modelo *feedforward* por suas características de aprendizado. Esses modelos são capazes de aprender padrões complexos e representações em grandes volumes de dados para realizar classificações. Além disso, esse modelo é menos complexo computacionalmente em comparação a outros, reduzindo custos de memória e sobrecarga (Haddadi *et al.*, 2010).

Durante o processo de treinamento de modelos, utilizamos algoritmos de otimização, como o Descida Gradiente Estocástico (SGD), que computam qual modificação de parâmetro

Figura 2 – Estrutura de Redes Neurais



Fonte: Elaborada pelo autor.

seria melhor para minimizar a função do treinamento e se aproximar do valor ideal da função baseada nos fatores taxa de aprendizado e função de custo (Ponti; Costa, 2018). A função de custo mede a diferença entre o valor real e o valor previsto para a função, enquanto a taxa de aprendizado define o tamanho das etapas necessárias para atingir o valor mínimo. Geralmente, uma taxa de aprendizado menor fornece melhor precisão na atualização de parâmetros (IBM, 2023).

O SGD é amplamente utilizado para minimizar a saída de uma função de perda através de etapas de ajustes com pequenos lotes de dados. Ele calcula um gradiente de erro a cada iteração e usa esse parâmetro para minimizar o erro na próxima etapa. Um gradiente é uma medida da taxa de variação de uma função em relação às suas variáveis, servindo para direcionar a função até o erro mínimo.

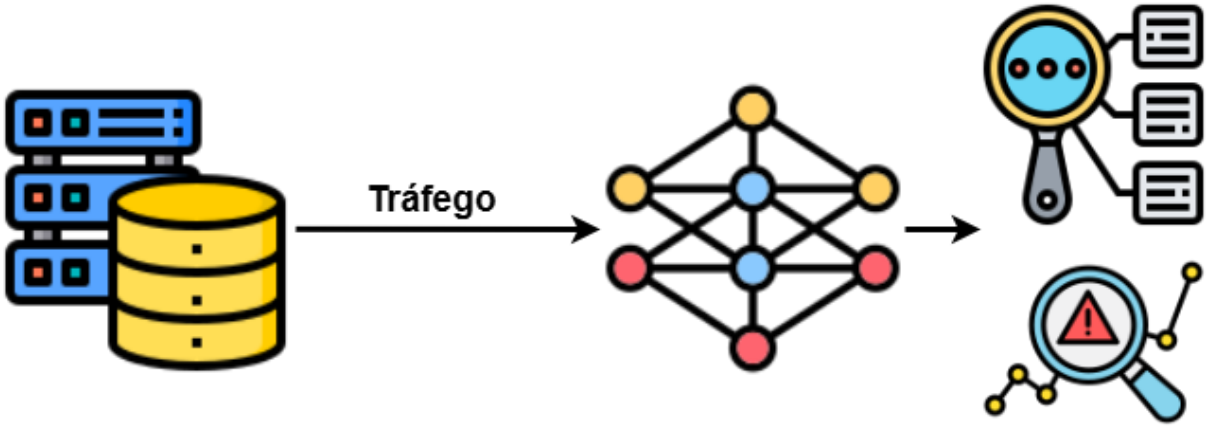
3.2.2 Classificação de Ataques

No contexto de dispositivos IoT, a classificação de ataques é uma técnica utilizada para identificar e classificar comportamentos maliciosos no tráfego de rede. Esses ataques podem ser originados de diversas fontes, como dispositivos comprometidos, vulnerabilidades nos protocolos de comunicação ou ações externas com a intenção de causar danos ao sistema (Damghani *et al.*, 2019). Em um ambiente IoT, os dados gerados pelos dispositivos podem ser

analisados para detectar padrões anômalos que indicam a presença de um ataque, com o objetivo de classificar a ameaça de forma precisa.

Métodos de classificação de ataques são utilizados como uma função de Sistemas de Detecção de Intrusão, junto da detecção de anomalias (Karatas *et al.*, 2018). Em um SDI, a classificação de ataques pode ser baseada em análise de assinaturas, que compara tráfego com características previamente conhecidas de ataques, e a detecção baseada em anomalias, que busca identificar desvios do comportamento normal. Algoritmos de *deep learning* são capazes de aprender com grandes volumes de dados e podem ser usados para treinar sistemas de detecção de intrusão baseados em anomalias, permitindo a identificação de padrões complexos e a distinção entre tráfego legítimo e malicioso com maior precisão. Um modelo classificador pode treinar com dados da rede e aprender a classificar o tráfego entre diferentes classes de ataque ou tráfego benigno (Ahmad *et al.*, 2021). Na Figura 3 vemos a representação de um modelo de redes neurais classificador de ataques, recebendo o tráfego e identificando atividades maliciosas ou não.

Figura 3 – Classificação de ataques com modelo de redes neurais



Fonte: Elaborada pelo autor.

3.2.3 TinyML

Tinyml é uma tecnologia emergente no contexto da inteligência artificial dedicada a integrar algoritmos de aprendizado de máquina a dispositivos de baixo custo e embarcados para funcionamento em ambientes críticos que vêm sendo ponto de interesse em diversas áreas, como saúde, computação e indústrias. As capacidades do *tinyml* permitem processamentos em tempo real com consumo de energia e de recursos computacionais, sendo ideal para ambientes com

dispositivos de recursos limitados que lidam com grandes quantidades de dados (Dutta; Bharali, 2021).

O uso de *tinyml* em ambientes limitados diminui o gargalo de processamento de dados na nuvem, o que aumenta a eficiência enquanto diminui a latência, tendo em vista que dispositivos finais são habilitados a processar fontes de dados locais para tomada de decisões ou retorno de *insights* com baixa latência e mínima utilização de recursos, além do fator de consumo energético baixo, que se voltam para aplicações com microcontroladores (Banbury *et al.*, 2020). Com essas características, os dispositivos realizam atividades de reconhecimento de áudio e imagem, manutenção preditiva e detecção de anomalias.

O Estado da Arte envolvendo *tinyml* para aplicações de detecção de anomalias aponta um foco em ambientes industriais, com algoritmos de aprendizado de máquina que identificam padrões em máquinas e equipamentos e geram alertas sobre funcionamentos inesperados, para evitar gastos com manutenções e entender melhor a situação real do ambiente (Siang *et al.*, 2021). Em contexto de redes IoT, o trabalho de (Dutta; Kant, 2021) aborda o uso de *tinyml* para implementação de modelos classificadores, como *Naive-Bayes*, para detecção de ameaças em conjuntos de dados.

O avanço do *tinyml* tem sido impulsionado pelo desenvolvimento de *frameworks* como *Edge Impulse* (Hymel *et al.*, 2022) e *Tensorflow Lite* (Tensorflow, 2021), especializados em treinar, otimizar e implementar os modelos de aprendizado de máquina e aprendizado profundo. O *Tensorflow Lite* possui integração com modelos baseados em *Tensorflow Keras* e fornece opções de otimização e quantização. Um modelo otimizado se torna compacto e eficiente enquanto mantém sua precisão. Processos mais rigorosos de otimização, como a quantização (Hubara *et al.*, 2021) alteram as configurações do modelo, convertendo seus parâmetros e operações. Um exemplo é mudar o padrão de operações de *float* 32 para *float* 16 ou *int* 8. Essa redução de *bits* diminui a complexidade das execuções, mas pode afetar a precisão do modelo (Tensorflow, 2024a).

O processo de quantização para reduzir modelos necessita de um conjunto de dados representativo. Os parâmetros do modelo em *float 32* são remapeados para outro valor (como *float* 16 ou *int* 8) com base nesse conjunto de dados. A redução de precisão é possível porque os valores em *float* 32 são escalonados para uma faixa menor de valores inteiros por meio de um fator de escala e um valor de deslocamento. O valor contínuo original é arredondado e mapeado para o inteiro mais próximo dentro da faixa definida. A perda de eficiência pode ocorrer devido

à mudança de precisão das operações, em que as representações que antes tinham uma faixa maior de valores numéricos decimais passam a ser definidas em uma faixa de valores discretos (redução de 32 *bits* para 8 *bits* (Krishnamoorthi, 2018).

3.3 Privacidade Diferencial

A privacidade diferencial (PD) é um modelo matemático robusto utilizado na computação para garantir a privacidade de um indivíduo em um conjunto de dados estatístico. O trabalho de (Dwork, 2006) introduziu o conceito de ε-diferencial, uma definição matemática para a perda de privacidade associada à exposição de dados em um conjunto de dados estatístico.

A privacidade diferencial pode ser definida matematicamente pela Inequação 3.1. Sendo $\mathscr{A}$ um algoritmo ou mecanismo aplicado aos conjuntos de dados D_1 e D_2, em que estes são chamados de vizinhos, pois se diferem em no máximo 1 elemento. Temos um número real positivo ε como parâmetro de privacidade e um conjunto S representando as saídas geradas pelo algoritmo $\mathscr{A}$ aplicado aos conjuntos de dados D_1 e D_2. Com esses elementos, consideramos o algoritmo $\mathscr{A}$ diferencialmente privado se a probabilidade do algoritmo, aplicado ao conjunto de dados D_1 gerar uma saída S, for menor ou igual à probabilidade de o mesmo mecanismo gerar uma saída S, mas com o conjunto de dados D_2 e com o fator de privacidade ε.

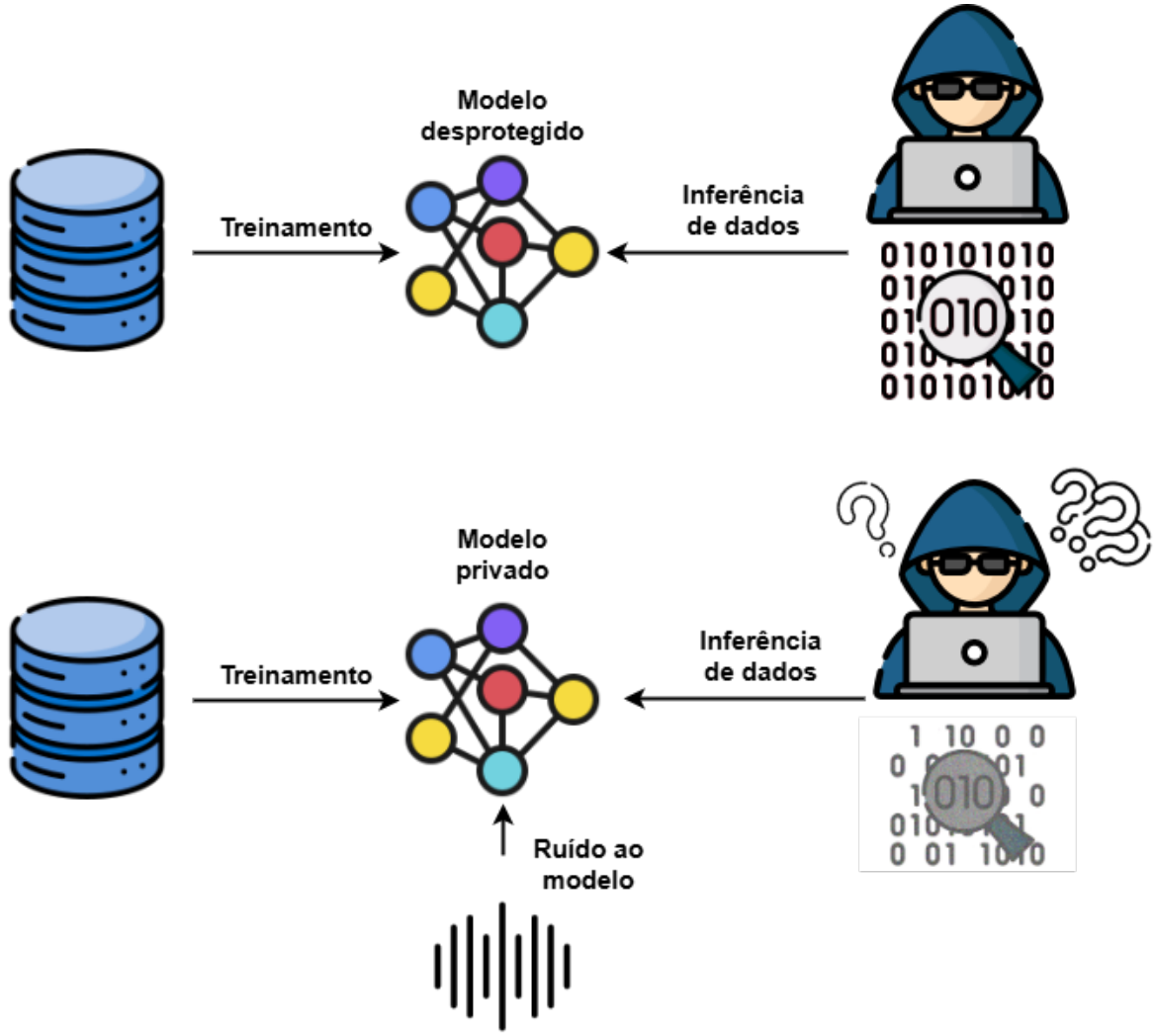
$$\Pr[\mathscr{A}(D_1) \in S] \leq e^{\varepsilon} \Pr[\mathscr{A}(D_2) \in S] \tag{3.1}$$

Com PD, conjuntos de dados e saídas de modelos de aprendizado de máquina obtêm um grau de ofuscação de informações através de ruído adicionado, mantendo a utilidade para uso geral (Wang *et al.*, 2023). A quantidade de ruído que adicionamos a um conjunto de dados afeta diretamente a utilidade dos dados. Quanto maior o valor de ε, mais utilidade e menos privacidade terão os dados. Por conta desse fator, devemos equilibrar a utilidade e privacidade exigidas para a finalidade dos dados.

Além de conjuntos de dados estatísticos, a PD é aplicável a contextos como fluxo de dados de IoT para agregadores de serviço (Vidal, 2020) e aprendizado profundo (Abadi *et al.*, 2016). Tornar o aprendizado profundo diferencialmente privado envolve garantir que o modelo não exponha dados usados no treinamento. Para isso, podemos adicionar ruído aos dados de treino, ofuscar as saídas do modelo ou usar um otimizador diferencialmente privado (Siachos *et al.*, 2023). Um otimizador diferencialmente privado manipula o processo de treinamento do

modelo com ruído e torna o modelo menos suscetível a vazamentos, como demonstra a Figura 4.

Figura 4 – Representação de modelo diferencialmente privado



Fonte: Elaborada pelo autor.

O otimizador Descida de Gradiente Estocástico possui uma variante diferencialmente privada: DP-SGD (Chua *et al.*, 2024), projetada para evitar que o modelo memorize características dos dados e diminuir o risco de exposição. O DP-SGD implementa rodadas de corte e adição de ruído. Tendo o limite de corte c, o *micro-batch* de tamanho B, sendo uma divisão ainda menor de um *mini-batch*, e o multiplicador de ruído α, o otimizador segue os seguintes passos a cada iteração: *i*) Para cada *mini-batch*, seleciona um *micro-batch* B aleatoriamente; *ii*) Calcula o gradiente de aprendizado; *iii*) Adiciona ruído de acordo com α; *iv*) Reduz o gradiente ao limite c; e *v*) Atualiza os parâmetros.

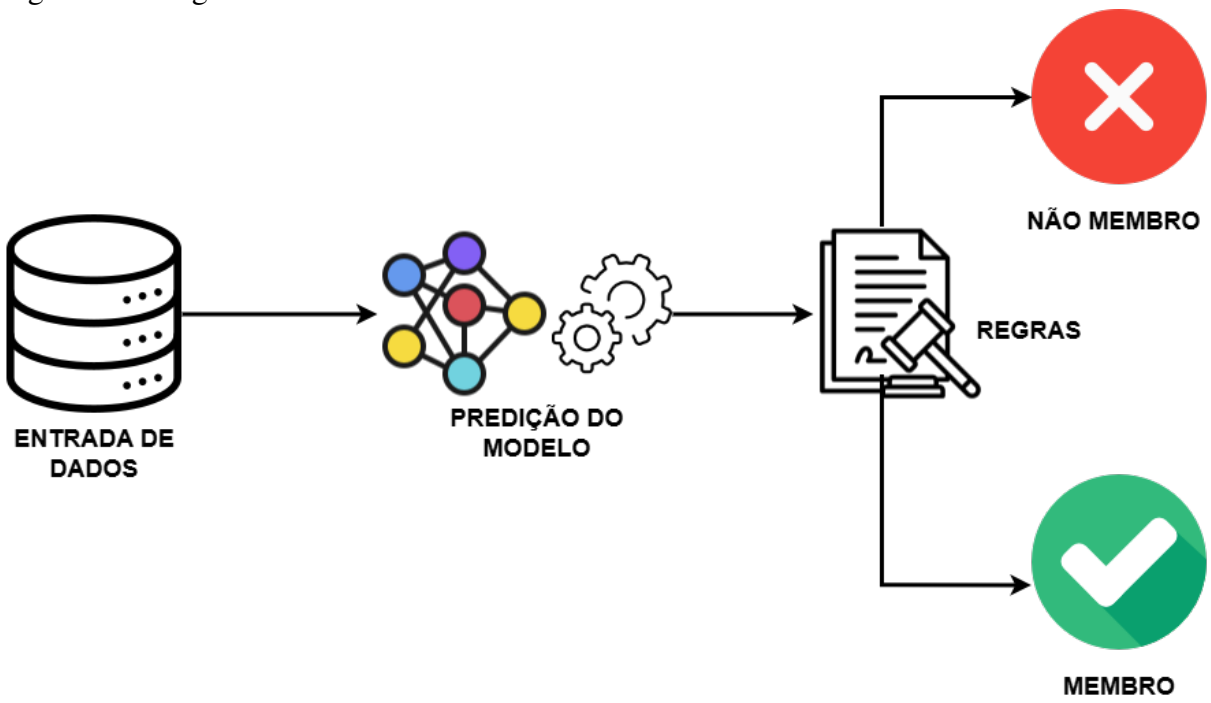
3.4 Ataque de Inferência de Associação

O Ataque de Inferência de Associação (MIA) é um tipo de ataque adversarial que se baseia em identificar se um registro de dado foi usado durante o treinamento de um modelo. Tendo acesso ao modelo e a uma amostra de dados, um atacante é capaz de utilizar suas predições para deduzir dados presentes no processo de treinamento (Hu *et al.*, 2022). Atacantes definem um modelo-alvo, que normalmente é disponibilizado publicamente, e o usam para gerar previsões em cima de uma amostra de dados. Conceitualmente, dados presentes no conjunto de treinamento são chamados de membros. Já os dados não presentes são chamados de não-membros. No contexto da IoT, um atacante com acesso a um modelo que atua com os dados da rede pode ter acesso às informações tanto dos usuários quanto dos dispositivos de rede, abrindo oportunidades para outros tipos de ataques, além dos riscos à privacidade.

Como exemplo, imagine que um grupo de pesquisa utiliza um conjunto de dados com informações médicas sensíveis sobre pacientes para treinar um modelo de classificação, publicando em seguida o modelo usado e seus resultados em um repositório *online* para fins de pesquisa. Um atacante, então, utiliza esse modelo e realiza um ataque de inferência para identificar amostras presentes nos dados de treinamento, podendo expor usuários e suas condições médicas (Zhang *et al.*, 2022).

Ataques de inferência de associação são classificados como ataques de caixa preta, quando o atacante tem acesso apenas ao modelo, e como ataques de caixa branca, quando há acesso às saídas, parâmetros e configurações do modelo (Wu *et al.*, 2023). Há uma variação chamada Ataque de caixa preta baseado em regras (Toolbox, 2018). Essa variação se baseia em identificar amostras utilizando predições do modelo, definindo membros com base em uma regra definida. Regras são instruções usadas para analisar os dados e inferir se são membros ou não. Seu funcionamento é demonstrado na Figura 5. Um exemplo de regra é definir uma amostra como membro se a predição do modelo for correta, com uma probabilidade acima de um limiar definido.

Figura 5 – Diagrama de funcionamento de MIA



Fonte: Elaborada pelo autor.

4 METODOLOGIA

Este capítulo visa apresentar as etapas de construção do trabalho proposto, com passo a passo de implementação e estudos de caso propostos.

4.1 Conjunto de Dados

4.1.1 Conjunto de Dados IoT-23

Definiu-se o conjunto de dados público IoT-23 (Garcia Agustin Parmisano, 2020), criado como parte do Laboratório *Avast AIC*, para treinamento e testes do modelo de rede neural. Esse conjunto de dados consiste em vinte e três capturas (cenários) de diferentes tipos de tráfego IoT, dos quais vinte cenários têm tráfego malicioso e três cenários têm tráfego benigno. A estratégia para o pré-processamento foi gerar um conjunto de dados com amostras de cinquenta mil pacotes de cada cenário. A coleta das amostras mantém a sequência temporal baseada no atributo *Timestamp* (Ts). O propósito foi obter um conjunto de dados com maior representação de ataques para verificar o desempenho do modelo proposto.

4.1.2 Pré-processamento

A tabela 2 demonstra 3 colunas apenas com os atributos disponíveis no conjunto de dados. Utilizou-se os seguintes atributos: Endereços *Internet Protocol* (IP) de origem e destino e portas de origem e destino, pois identificam os pontos finais de comunicação e descrevem a comunicação da rede, dispositivos e processos, sendo indicadores para atividades maliciosas. Além disso, escolheu-se o número de *bytes* de pacotes para caracterização e protocolo usado para identificar a comunicação. O atributo *label* define o rótulo do tráfego entre benigno ou em um dos ataques. Após a seleção dos atributos, o arquivo final ficou com os atributos *ts*, *id.orig h*, *id.orig p*, *id.resp h*, *id.resp p*, *proto*, *orig ip bytes* e *label*.

A Tabela 3 resume os rótulos presentes no conjunto de dados inicialmente e suas quantidades. Descartou-se do conjunto de treinamento as classes *C&C-HeartBeat*, *C&C-FileDownload*, *C&C-HeartBeat-FileDownload*, *attack*, *C&C-Torii*, *FileDownload* e *C&C-Mirai*. Essas classes possuem poucas amostras e são variantes dos ataques selecionados. Apesar da implementação de balanceamento de dados, descartamos essas classes por gerarem muitas réplicas pouco diversificadas, o que é responsável por gerar *overfitting* e diminuir a generalização do

Tabela 2 – Atributos do conjunto de dados

Atributos	Atributos	Atributos
ts	service	missed bytes
uid	duration	history
id.orig h	orig bytes	org pkts
id.orig p	resp bytes	orig ip bytes
id.resp h	conn state	resp pkts
id.resp p	local orig	resp ip bytes
proto	local resp	label

Fonte: Elaborada pelo autor.

modelo de rede neural. Aplicou-se a técnica de balanceamento *Synthetic Minority Over-sampling Technique* (SMOTE) (Chawla *et al.*, 2002) para gerar amostras sintéticas das classes de dados minoritárias com base em características de vizinhos próximos e balancear proporcionalmente o conjunto de dados. A Tabela 4 indica as classes selecionadas e balanceadas para o treinamento do modelo. Aplicou-se a transformação de valores categóricos em numéricos com a técnica *Label Encoder* no conjunto de dados e a técnica de normalização *MinMaxScaler* para normalizar os valores das amostras de dados num intervalo entre 0 e 1. Esse pré-processamento facilita o treinamento do modelo. A divisão do conjunto de dados foi estabelecida em 50% para treinamento, 30% para teste e 20% para validação.

Tabela 3 – Rótulos e suas quantidades

Rótulo	Quantidade	Rótulo	Quantidade
benigno	109177	*PartOfAHorizontal PortScan*	413024
C&C	15064	*Attack*	1387
C&C-HeartBeat	229	*Okiru*	131308
DDoS	76371	*C&C-Torii*	30
C&C-FileDownload	44	*FileDownload*	16
C&C-HeartBeat-FileDownload	11	C&C-Mirai	1

Fonte: elaborada pelo autor.

Tabela 4 – Rótulos selecionados após balanceamento

Rótulo	Quantidade
benigno	104446
DDoS	104220
Okiru	104287
C&C	104298
PartOfAHorizontalPortScan	104209

Fonte: Elaborada pelo autor.

4.2 Privacidade Diferencial para Aprendizado Profundo

Para este projeto, a solução é implementada baseada no algoritmo Estimativa de Momento Adaptativa Diferencialmente Privada (DP-ADAM) (Tang *et al.*, 2024), uma variação do DP-SGD que possui o mesmo funcionamento e objetivo, porém utiliza estimativas de momento, como média e variância, para atualizar os parâmetros de treinamento. O DP-ADAM limita os gradientes e adiciona ruído aleatório para impedir que o modelo memorize características dos dados, reduzindo o risco de exposição. Esse algoritmo é um otimizador que torna o modelo diferencialmente privado por meio de ruído no processo de aprendizado do modelo.

Utilizou-se a biblioteca *tensorflow privacy* (Tensorflow, 2024c) para implementar o otimizador DP-ADAM integrado à API *Keras* (Chollet *et al.*, 2015). Esse otimizador atua diretamente no treinamento e seus parâmetros determinam o nível de privacidade do modelo. Seus parâmetros servem para, durante uma iteração do treinamento, separar os gradientes em lotes de tamanho fixo *n*, adicionar um nível de ruído e limitar o tamanho do gradiente de acordo com um limite definido (privacy, 2024).

No experimento, utilizamos os parâmetros Multiplicador de ruído e Norma de corte como fatores para impactar diretamente no desempenho do modelo. A norma de corte é um parâmetro que limita a sensibilidade dos gradientes de treinamento, limitando seus valores após a adição de ruído para que nenhum ponto de dados se sobreponha a outros e influencie o modelo de forma individual. O Multiplicador de ruído é um parâmetro com valores entre 0 e $\infty+$ que serve como um fator de adição de ruído para os gradientes de treino. O valor de *epsilon* ε é definido por uma função do *tensorflow privacy* que recebe os parâmetros de treinamento do modelo, os parâmetros do otimizador DP-ADAM e retorna o nível de privacidade ε. Utilizamos 3 valores de norma de corte (1.2, 1.5 e 1.7), e para cada norma de corte, utilizamos 7 valores de *epsilon* ε: (1.56, 1.0, 0.6, 0.4, 0.29, 0.12 e 0.08).

4.3 Modelo de Classificação

4.3.1 Classificador FeedForward

A solução foi desenvolvida baseada no modelo de redes neurais profundas do tipo *FeedForward* (Rimer; Martinez, 2004). Escolheu-se essa arquitetura por ser capaz de realizar tarefas de classificação com uma estrutura menos complexa e que gera menos consumo computa-

cional. Apesar da simplicidade, o modelo aprende as características dos dados e generaliza bem, garantindo alta taxa de acerto na classificação. Utilizou-se a função *Softmax* (Qi *et al.*, 2017) para determinar a probabilidade de pertencimento do valor de entrada a cada uma das classes dos dados.

Implementou-se o modelo utilizando a *API Keras*, do *TensorFlow*. Adotou-se uma estrutura de 5 camadas totalmente conectadas (densas) com poucos nós de processamento. Aplicou-se a função *softmax* na camada de saída e a função de ativação Unidade Linear Retificada (ReLU) nas camadas de entrada e ocultas. A estrutura do modelo pode ser vista na Figura 6, com as camadas de entrada e saída de dados conectadas às camadas ocultas, em que cada camada tem um número de nós definidos e uma função de ativação. A Tabela 5 resume os parâmetros de arquitetura e de treinamento.

Tabela 5 – Parâmetros do modelo *baseline*

Parâmetro	Argumento
Tipo	Sequencial; Denso
Função de ativação	ReLU, Softmax
Otimizador	ADAM
Função de perda	*Sparse Categorical Crossentropy*
Épocas	20
batch size	128

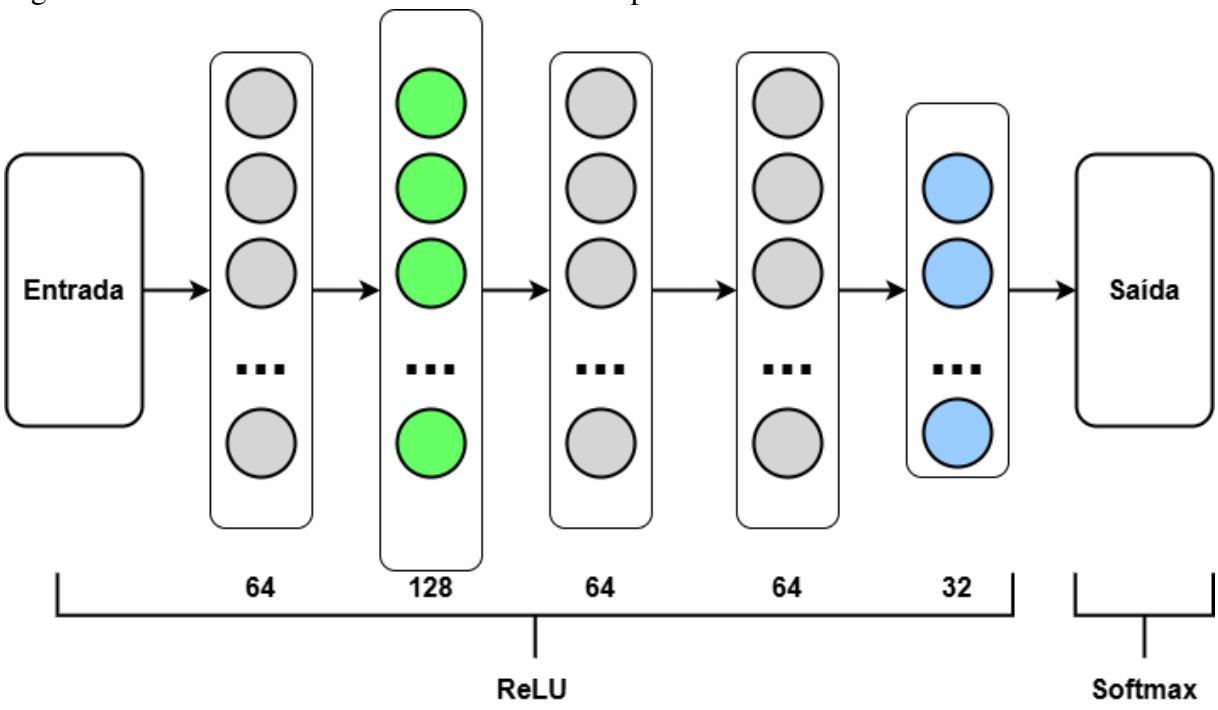
Fonte: Elaborada pelo autor.

A construção de modelo mostrada na Tabela 5 e Figura 6 é a arquitetura básica utilizada nos experimentos. Destacamos um modelo *baseline* com essas características e comparamos com os experimentos de privacidade diferencial, criando 21 versões desse modelo com diferentes parâmetros. Tendo 3 valores de norma de corte e 7 valores de ε, cada modelo possui um nível de privacidade diferente a ser comparado.

Para avaliação de desempenho, utilizamos as métricas acurácia, precisão, *recall* e *f1-score*, demonstradas nas Equações 4.1, 4.2, 4.3 e 4.4. Essas métricas são definidas usando a matriz de confusão do modelo, que informa o número de *Verdadeiros Positivos* (VP), *Falsos Positivos* (FP), *Verdadeiros Negativos* (VN) e *Falsos Negativos* (FN). Com esses valores, as métricas de desempenho são definidas como:

$$\text{Acurácia} = \frac{VP+VN}{VP+VN+FP+FN} \tag{4.1}$$

Figura 6 – Estrutura do modelo de redes neurais profundas



Fonte: Elaborada pelo autor.

$$\text{Precisão} = \frac{VP}{VP+FP} \tag{4.2}$$

$$\text{Recall} = \frac{VP}{VP+FN} \tag{4.3}$$

$$\text{F1-score} = 2 \times \frac{\text{Precisão} \times \text{Recall}}{\text{Precisão} + \text{Recall}} \tag{4.4}$$

4.4 Implementação de modelo *tinyml*

Utilizou-se o conceito de *tinyml* para otimizar nosso modelo de classificação de ataques. Esse modelo otimizado tem funcionamento centralizado para redes IoT e consome menos memória e custo computacional, além de realizar inferências em fluxos de dados em um período de tempo menor. O processo de *tinyml* pode ser implementado a partir de *frameworks* para redução de modelos. Utilizamos o *Tensorflow Lite* (Tensorflow, 2021) para otimizar o modelo proposto. Essa biblioteca possui integração com modelos baseados em *Tensorflow Keras* e fornece funções para otimizar o modelo em diversos níveis.

Realizamos a otimização padrão fornecida pelo *tensorflow lite* e comparamos o tamanho final do modelo, seu tempo de execução e consumo de memória. Isso é feito com o modelo base salvo em formato .h5 (extensão para *tensorflow keras*) e modelo otimizado em formato .tflite (extensão para o modelo reduzido). Utilizamos um conjunto de teste pré-processado com 223484 amostras e cada modelo realizou uma inferência a todas as amostras. A Tabela 6 resume os resultados dessa comparação. Vemos que o modelo otimizado realiza inferências com um consumo de memória e tempo baixos, enquanto ocupa alguns *bytes* de armazenamento, ideal para ambientes com grandes fluxos de dados e menor poder computacional.

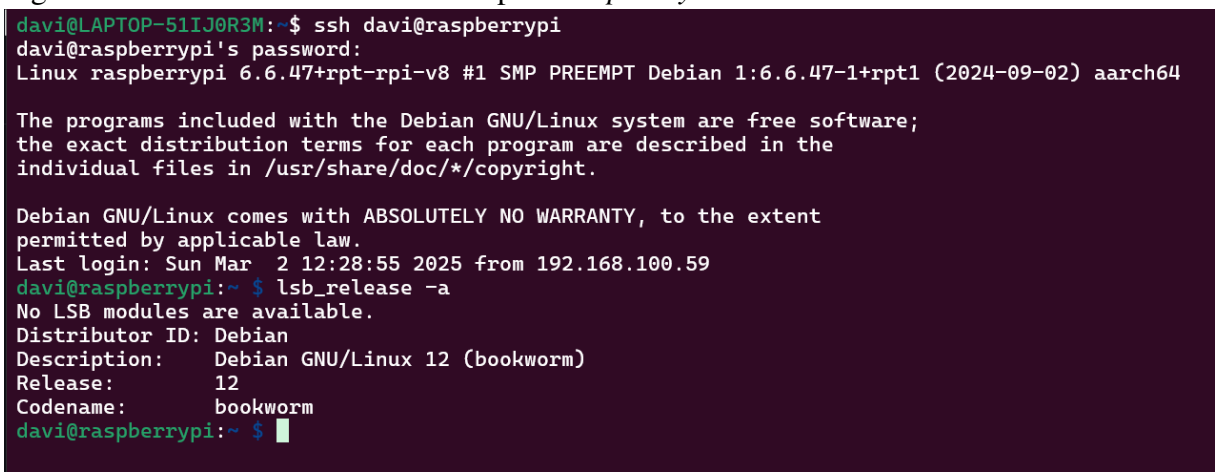
Tabela 6 – Comparação entre modelo base e otimizado

Modelo	Tamanho (MB)	Tempo de Execução (s)	Consumo de Memória (MB)
Modelo Base	0.32	34.0142	8.35
Modelo Otimizado	0.09	3.2141	0.05

Fonte: Elaborada pelo autor.

Utilizou-se o dispositivo *Raspberry Pi 4 Model B* (Raspberrypi, 2024), um microcomputador de capacidade computacional maior em relação a microcontroladores. Essa capacidade permite um nível de otimização menor para o modelo classificador de ataques. Utilizamos o *Raspbian OS* de 64 *bits* no dispositivo, um sistema operacional baseado em *Linux* (Raspberrypi, 2024). No sistema, implementamos o ambiente de funcionamento do modelo com a biblioteca *tflite_runtime* (Pypi, 2023), responsável pelo funcionamento do modelo otimizado. O carregamento do modelo e a execução do ambiente são feitos via SSH. A Figura 7 ilustra o acesso feito via terminal no ambiente do dispositivo *Raspberry Pi*.

Figura 7 – Conexão SSH via terminal para *Raspberry Pi*

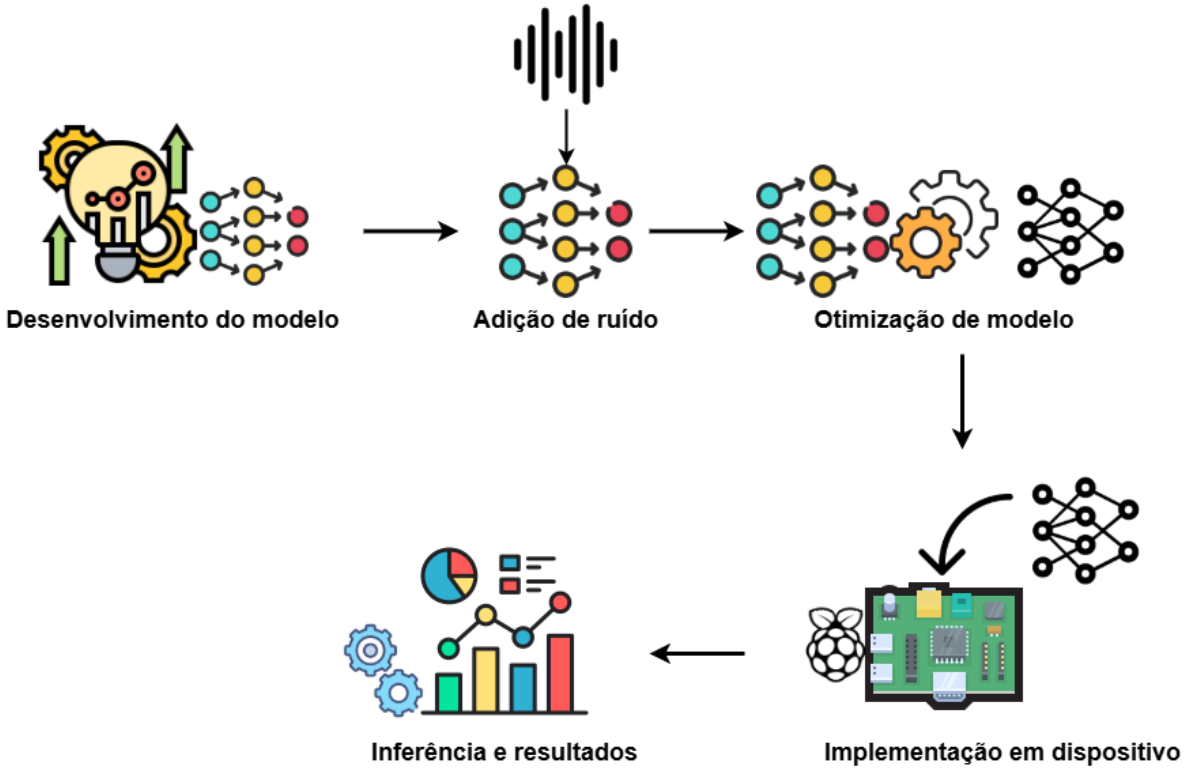


Fonte: Elaborada pelo autor.

Quando reduzido, o modelo otimizado funciona com interpretadores *tensorflow*

(Tensorflow, 2024b). Eles são componentes que executam modelos em formato .tflite e interpretam de forma eficiente a entrada (amostras) e saída (resultado da inferência) de dados do modelo através de tensores gerenciados. Cada modelo possui um tensor de entrada e um de saída com metadados de execução que informam o tipo de dados, formato e informações de quantização. O interpretador gerencia esses tensores e aloca os dados de execução para obter mais eficiência.

Figura 8 – Etapas e ambientes de modelagem



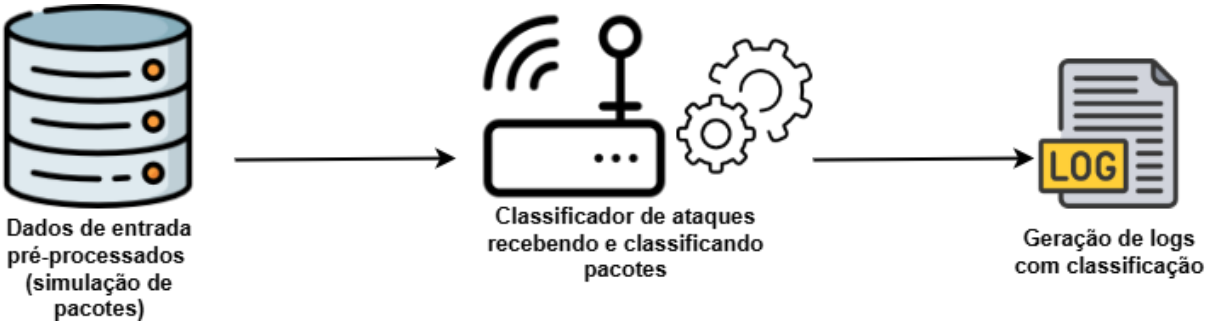
Fonte: Elaborada pelo autor.

A Figura 8 mostra as principais etapas do processo de implementação. A otimização é a etapa de converter o sistema classificador de ataques para funcionamento em baixo nível. Em um primeiro experimento, escolheu-se o modelo diferencialmente privado treinado sob norma de corte = 1.5 e $\varepsilon = 0.6$. Suas métricas de desempenho podem ser vistas no Capítulo 5. Sua versão otimizada mantém as mesmas métricas. Utilizamos o modelo diferencialmente privado escolhido para comparar entre duas formas de otimização, sendo a otimização padrão e a quantização *int 8* (Kim *et al.*, 2021). Ao realizarmos uma quantização inteira de 8 *bits*, mudamos o formato dos parâmetros e funções do modelo de *float* 32 para *int* 8, o que aumenta o desempenho em dispositivos de baixo custo, enquanto afeta a capacidade de classificação. Os resultados da comparação feita são demonstrados no Capítulo 5.

Como estudo de caso, implementou-se um *pipeline* reduzido de detecção de ataques

para simular um processo de captura e classificação de tráfego de rede. Nesse experimento, o sistema implementado no *Raspberry Pi* simula um funcionamento em tempo real de captura de pacotes e classificação de tráfego, utilizando pacotes já pré-processados. A Figura 9 demonstra o cenário do experimento, com a recepção dos dados, classificação e geração de *logs*.

Figura 9 – Cenário de experimento



Fonte: Elaborada pelo autor.

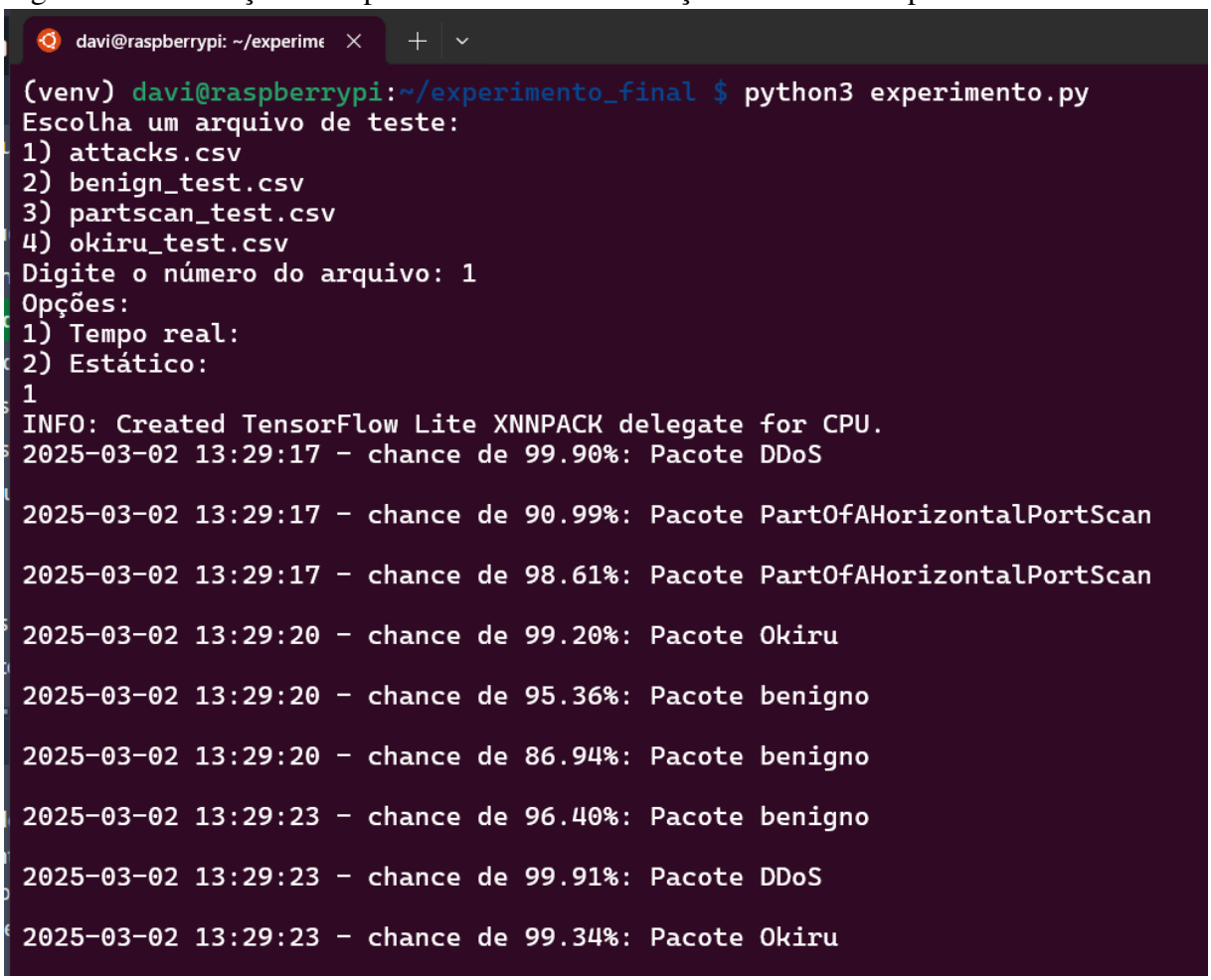
O modelo fica em modo de escuta e recebe de 1 a 6 amostras aleatórias a cada período de tempo entre 1 e 3 segundos. O sistema captura esses dados e classifica entre as classes aprendidas, gerando arquivos de *log* para cada captura realizada. Utilizamos quatro conjuntos de teste para esse experimento, chamados *attacks.csv*, *benign.csv*, *partscan.csv* e *okiru.csv*. O primeiro consiste no conjunto de teste com todas as classes e 223484 amostras. Os outros três são derivações obtidas do pré-processamento, com foco em alguma classe específica do conjunto de dados IoT-23. O arquivo *benign.csv* possui amostras de tráfego benigno, enquanto o *partscan.csv* possui amostras com foco em *DDoS* e *PartOfAHorizontalPortScan*. O arquivo *okiru.csv* tem pacotes *okiru*. As quantidades de amostras de cada arquivo são demonstradas na Tabela 7. Um exemplo de execução pode ser visto na Figura 10, com a classificação feita no conjunto de teste *attacks.csv* em tempo real. Os arquivos de *log* gerados são armazenados com o registro de tempo e data da captura.

Tabela 7 – Arquivos de teste

Arquivo	Quantidade
attacks.csv	223484
benign.csv	104223
partscan.csv	208616
okiru.csv	104195

Fonte: Elaborada pelo autor.

Figura 10 – Execução do experimento com classificação feita em cada pacote



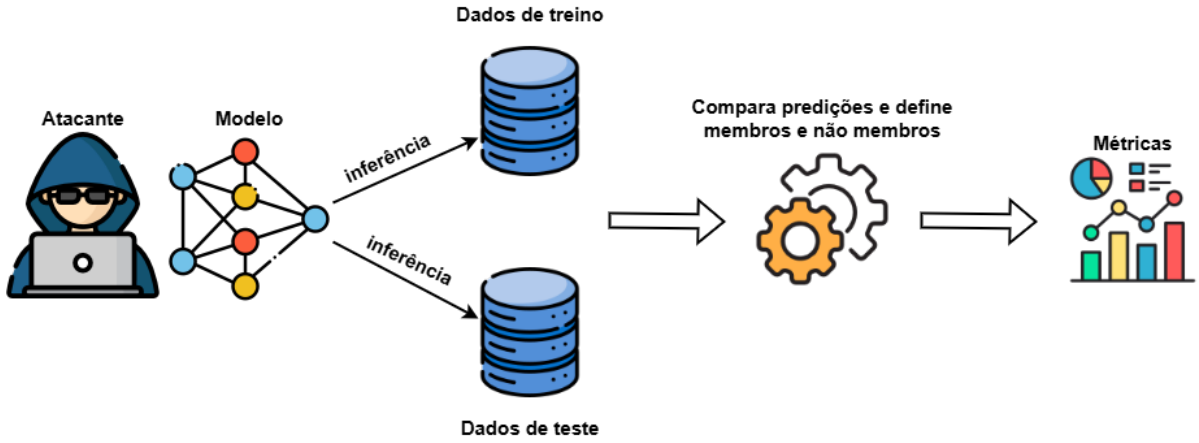
Fonte: Elaborada pelo autor.

4.5 Cenário de Ataque de Inferência de Associação

A implementação do ataque de inferência de associação baseado em regras está disponível em (AI, 2024). No algoritmo utilizado, o ataque tem acesso ao modelo e aos conjuntos de treino e teste, gerando um valor de predição para cada entrada. O ataque implementado usa uma regra básica, em que o modelo mantém uma lista com membros e não-membros, representados por 1 e 0, respectivamente. Nesse cenário, definimos membros como o conjunto de treino e não-membros como conjunto de teste. Ao receber uma entrada de exemplo, o modelo realiza uma predição nos conjuntos de treino e teste, e se uma predição for considerada correta, a amostra recebe o valor 1 e é considerada um membro. Caso contrário, recebe valor 0 e é considerada não-membro. A Figura 11 mostra o funcionamento do ataque. As classificações realizadas são demonstradas no Capítulo 5.

Aplicaram-se as seguintes métricas: acurácia de membros (*member acc*), acurácia de não membros (*non-member acc*), acurácia do ataque (*attack acc*), precisão e *recall*. Essas

Figura 11 – Funcionamento do ataque de inferência baseado em regras



Fonte: Elaborada pelo autor.

métricas são definidas nas Equações 4.5, 4.6, 4.7, 4.8 e 4.9.

$$\textit{member acc} = \frac{\text{membros corretamente inferidos}}{\text{total de membros}} \tag{4.5}$$

$$\textit{non-member acc} = 1 - \frac{\text{não membros inferidos como membros}}{\text{total de não membros}} \tag{4.6}$$

$$\textit{attack acc} = \frac{\text{member acc} \times \text{No. de membros} + \text{non-member acc} \times \text{No. de não membros}}{\text{total de amostras}} \tag{4.7}$$

$$\text{precisão} = \frac{\text{No. de positivos previstos corretamente}}{\text{total de positivos previstos}} \tag{4.8}$$

$$\textit{recall} = \frac{\text{No. de positivos previstos corretamente}}{\text{total de positivos reais}} \tag{4.9}$$

5 RESULTADOS

Este capítulo reúne os resultados obtidos nos experimentos realizados. A Seção 5.1 demonstra os resultados obtidos de avaliações realizadas com o modelo classificador *baseline*. Em seguida, a Seção 5.2 compara os resultados obtidos dos experimentos com versões diferencialmente privadas do classificador de ataques. Por fim, a Seção 5.3 resume os resultados alcançados em experimentos com o modelo otimizado.

5.1 Modelo *baseline*

o modelo *baseline* demonstrou uma alta taxa de acurácia, atingindo um desempenho médio de 99%, com uma matriz de confusão que evidenciou poucas falhas na classificação das classes 0 e 4, o que é evidenciado nas Tabelas 8 e 9. No entanto, essa alta performance também resultou em uma maior vulnerabilidade ao ataque de inferência de associação (MIA), que obteve uma acurácia de 70% (Tabela 10), indicando que o modelo original poderia expor informações sensíveis dos dados de treinamento.

A Figura 12 ilustra a curva de perda do modelo *baseline*, mostrando que o treinamento não apresentou *overfitting*, mas demonstrou dificuldades de ajuste nas últimas épocas (iterações de treinamento para os mesmos dados). Isso sugere que o modelo conseguiu generalizar bem os padrões do conjunto de dados, embora tenha tido desafios específicos na convergência final.

Tabela 8 – Métricas do modelo *baseline*

Acurácia	Precisão	*Recall*	*f1-Score*
0.99	0.98	0.99	0.99

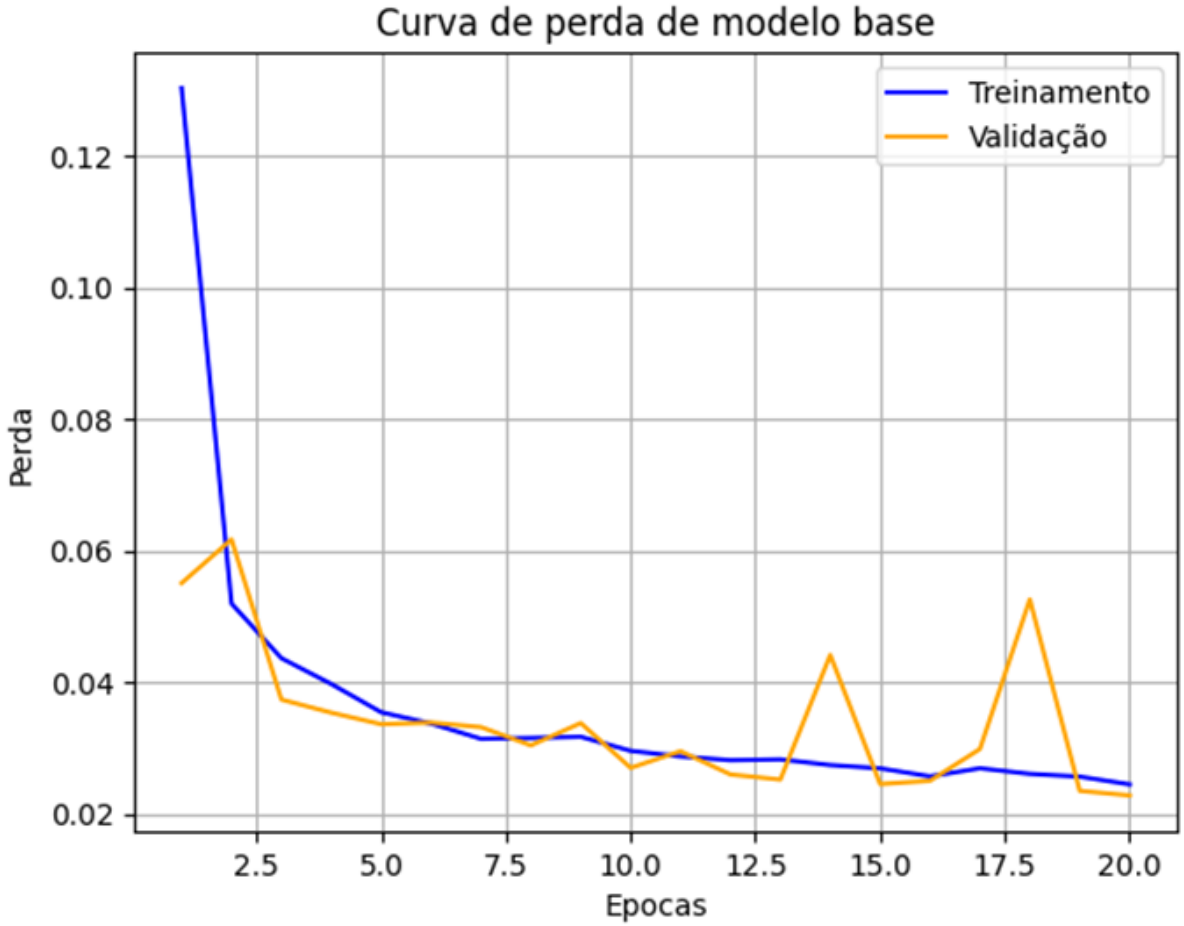
Fonte: Elaborada pelo autor.

Tabela 9 – Matriz de confusão para o modelo *baseline*

Classes	Prevista 0	Prevista 1	Prevista 2	Prevista 3	Prevista 4
Real 0	31759	36	11	1	954
Real 1	4	4442	0	0	1
Real 2	29	0	22810	0	0
Real 3	1	0	0	39311	0
Real 4	595	196	0	0	123334

Fonte: Elaborada pelo autor.

Figura 12 – Curva de perda de treinamento e validação do modelo *baseline*s



Fonte: Elaborada pelo autor.

Tabela 10 – Métricas de ataque em cima do modelo *baseline*

member acc	*non-member acc*	*attack acc*	precisão	*recall*
0.99	0.008	0.7	0.7	0.99

Fonte: Elaborada pelo autor.

5.2 Modelos diferencialmente privados

Tendo os valores de norma de corte: *i*) 1.2, *ii*) 1.5 e *iii*) 1.7; e os níveis de privacidade (ε): *i*) 0.08, *ii*) 0.12, *iii*) 0.29, *iv*) 0.4, *v*) 0.6, *vi*) 1.0, *vii*) 1.56, temos 21 versões diferencialmente privadas do modelo *baseline*. Essa seção reúne o desempenho de classificação desses modelos e o desempenho do ataque MIA a cada modelo. Os resultados são demonstrados nas Tabelas 11, 13 e 15.

Com a introdução da privacidade diferencial, observou-se um declínio progressivo na acurácia do modelo à medida que ε diminuía. As Tabelas 11, 13 e 15 demonstram essa relação para normas de corte de 1.2, 1.5 e 1.7, respectivamente. Para ε = 1.56, o modelo ainda manteve

Tabela 11 – Métricas dos modelos para norma de corte = 1.2

ε	acurácia	precisão	*f1-score*	*recall*
1.56	0.91	0.83	0.87	0.95
1.0	0.90	0.83	0.87	0.95
0.6	0.92	0.85	0.89	0.95
0.4	0.88	0.80	0.84	0.94
0.29	0.85	0.74	0.79	0.91
0.12	0.71	0.65	0.70	0.86
0.08	0.62	0.57	0.60	0.76

Fonte: Elaborada pelo autor.

Tabela 12 – Métricas de ataque para norma de corte = 1.2

ε	*member acc*	*non-member acc*	*attack acc*	*recall*
1.56	0.90	0.09	0.66	0.90
1.0	0.90	0.09	0.66	0.90
0.6	0.91	0.08	0.66	0.91
0.4	0.87	0.12	0.65	0.87
0.29	0.85	0.15	0.64	0.85
0.12	0.71	0.28	0.58	0.71
0.08	0.62	0.37	0.54	0.62

Fonte: Elaborada pelo autor.

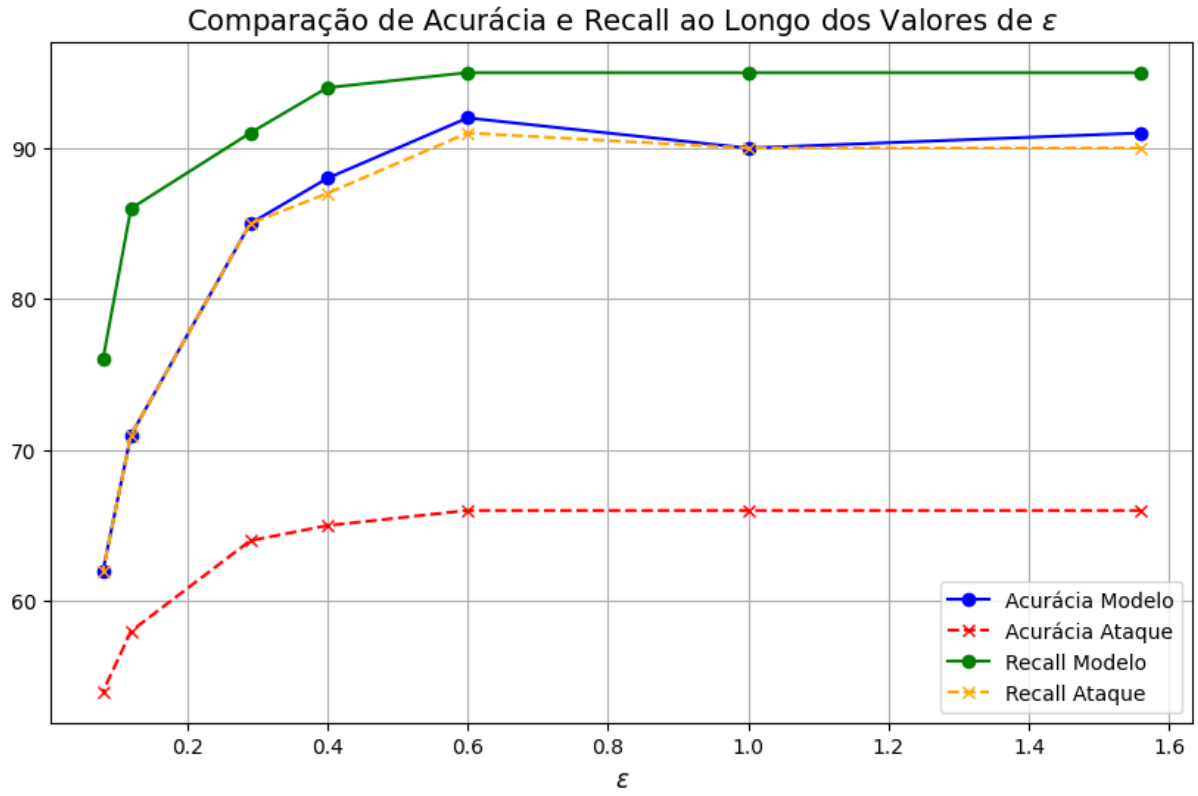
uma acurácia superior a 90%, mas quando ε foi reduzido para 0.08, a acurácia caiu para cerca de 62% na norma de corte 1.2, 64% na norma de corte 1.5 e 56% na norma de corte 1.7. Esses resultados evidenciam o *trade-off* inerente entre privacidade e utilidade: à medida que mais ruído é adicionado ao modelo, sua capacidade de classificação diminui, impactando diretamente na detecção de ataques.

Além disso, os ataques de inferência foram significativamente afetados pela introdução da privacidade diferencial. A Tabela 12 mostra que, para a norma de corte de 1.2 e $\varepsilon = 0.08$, a acurácia do ataque foi reduzida para 54%, representando uma queda de 16 pontos percentuais em relação ao modelo *baseline*. Tendências semelhantes foram observadas nas Tabelas 14 e 16 para normas de corte 1.5 e 1.7, reforçando que a aplicação de privacidade diferencial reduz a eficácia do ataque de inferência, mesmo ao custo de uma menor capacidade de classificação do modelo.

A Figura 13 compara graficamente a acurácia do modelo com o *recall* do ataque para norma de corte 1.2, demonstrando que o desempenho do ataque se mantém proporcional ao modelo. Esse comportamento também é visto nas Figuras 14 e 15 para normas de corte 1.5 e 1.7, respectivamente, reforçando a tendência de que o ataque se torna menos eficaz à medida que o nível de privacidade aumenta. O *recall* do ataque é próximo à acurácia do modelo porque o ataque de inferência de associação se baseia na confiança das predições para distinguir amostras

do conjunto de treinamento. Como modelos mais precisos tendem a gerar previsões com maior confiança para amostras vistas no treino, o ataque consegue identificar corretamente esses casos com uma taxa proporcional à acurácia do modelo. Assim, a taxa de verdadeiros positivos do ataque se aproxima da taxa de classificações corretas do modelo.

Figura 13 – Comparação entre acurácia e *recall* de modelo e ataque para norma de corte = 1.2



Fonte: Elaborada pelo autor.

Tabela 13 – Métricas dos modelos para norma de corte = 1.5

ε	acurácia	precisão	*f1-score*	*recall*
1.56	93	84	88	96
1.0	90	84	88	95
0.6	88	80	84	94
0.4	82	73	78	91
0.29	86	76	81	94
0.12	64	58	61	81
0.08	64	55	57	68

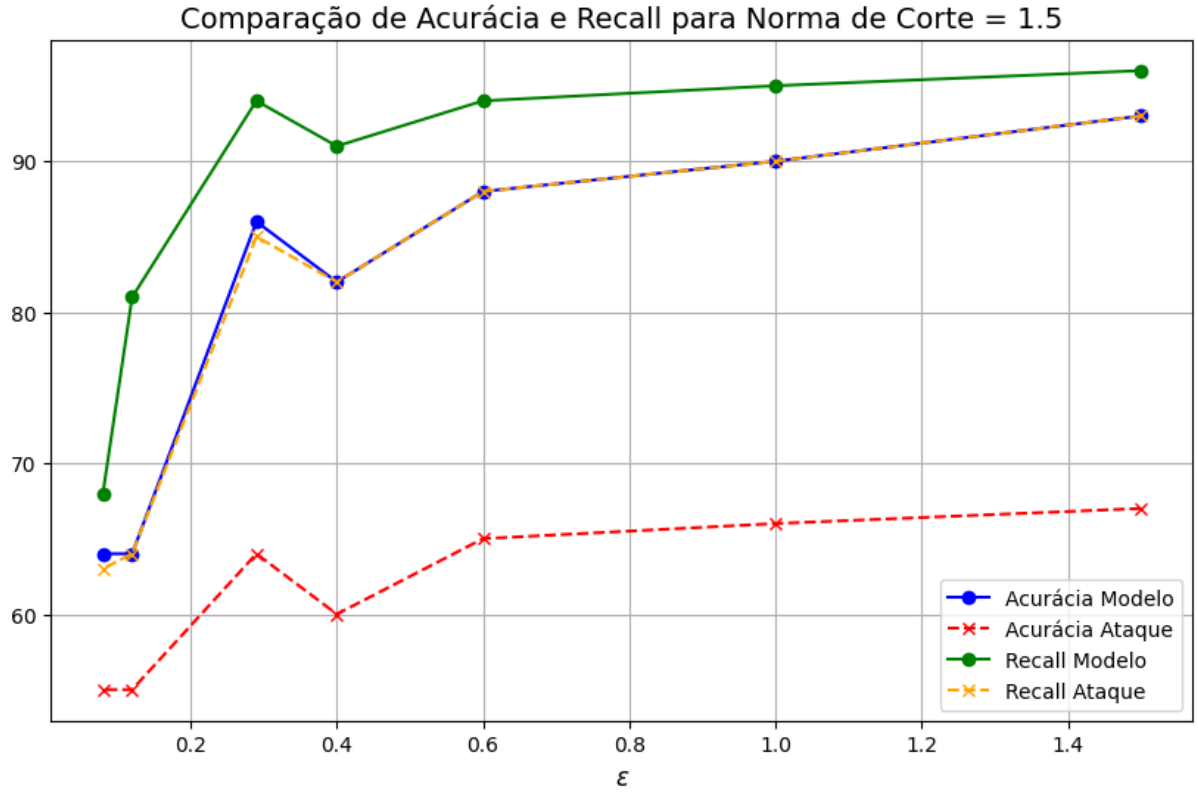
Fonte: Elaborada pelo autor.

Tabela 14 – Métricas de ataque para norma de corte = 1.5

ε	*member acc*	*non-member acc*	*attack acc*	*recall*
1.56	0.93	0.07	0.67	0.93
1.0	0.90	0.09	0.66	0.90
0.6	0.88	0.12	0.65	0.88
0.4	0.82	0.17	0.60	0.82
0.29	0.85	0.14	0.64	0.85
0.12	0.64	0.35	0.55	0.64
0.08	0.63	0.36	0.55	0.63

Fonte: Elaborada pelo autor.

Figura 14 – Comparação entre acurácia e *recall* de modelo e ataque para norma de corte = 1.5



Fonte: Elaborada pelo autor.

Tabela 15 – Métricas dos modelos para norma de corte = 1.7

ε	acurácia	precisão	*f1-score*	*recall*
1.56	90	82	86	95
1.0	89	83	87	94
0.6	86	76	80	93
0.4	86	79	84	94
0.29	83	74	74	91
0.12	70	61	65	82
0.08	56	57	57	79

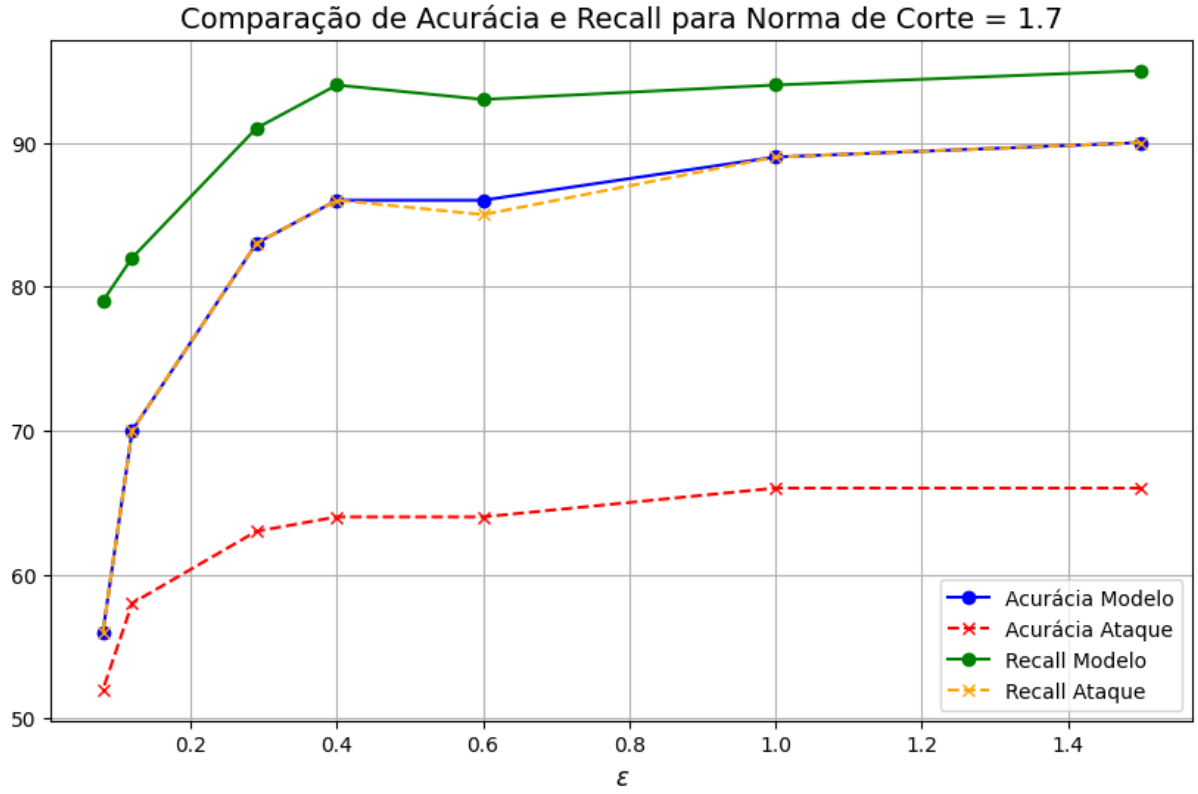
Fonte: Elaborada pelo autor.

Tabela 16 – Métricas de ataque para norma de corte = 1.7

ε	*member acc*	*non-member acc*	*attack acc*	*recall*
1.56	0.89	0.10	0.66	0.90
1.0	0.89	0.10	0.66	0.89
0.6	0.85	0.14	0.64	0.85
0.4	0.86	0.13	0.64	0.86
0.29	0.83	0.17	0.63	0.83
0.12	0.70	0.29	0.58	0.70
0.08	0.56	0.43	0.52	0.56

Fonte: Elaborada pelo autor.

Figura 15 – Comparação entre acurácia e *recall* de modelo e ataque para norma de corte = 1.7



Fonte: Elaborada pelo autor.

5.3 Modelo *tinyml*

Esta seção resume os resultados obtidos nos experimentos com os modelos otimizados. Os experimentos foram definidos na metodologia e têm o objetivo de demonstrar o uso de modelos diferencialmente privados de forma otimizada.

O primeiro experimento é uma comparação feita entre duas formas de otimização com o modelo diferencialmente privado com norma de corte = 1.5 e ε = 0.6. A Tabela 17 mostra o impacto da otimização padrão e da quantização inteira de 8 *bits* no mesmo modelo. A otimização padrão manteve o desempenho apresentado na Tabela 13, enquanto a quantização reduziu a precisão do modelo.

A Tabela 18 traz uma comparação entre o tamanho, tempo de execução e consumo de memória entre o modelo básico, o modelo com otimização padrão e o modelo com quantização inteira de 8 *bits*. Essa comparação mostra que a otimização padrão é mais eficiente em termos de consumo computacional, com valores baixos para tempo de execução e consumo de memória.

Tabela 17 – Comparação de desempenho entre otimização padrão e quantização

Métrica	Padrão	Quantização *int 8*
Acurácia	0.88	0.85
Precisão	0.80	0.78
Recall	0.94	0.93
f1-Score	0.84	0.82

Fonte: elaborada pelo autor.

Tabela 18 – Comparação entre modelo base, otimização padrão e quantização inteira de 8 *bits*

Modelo	Tamanho (MB)	Tempo de Execução (s)	Consumo de Memória (MB)
Modelo Base	0.32	20.5738	6.23
Modelo Otimizado	0.09	1.8884	0.05
Modelo Quantizado	0.03	8.8276	0.06

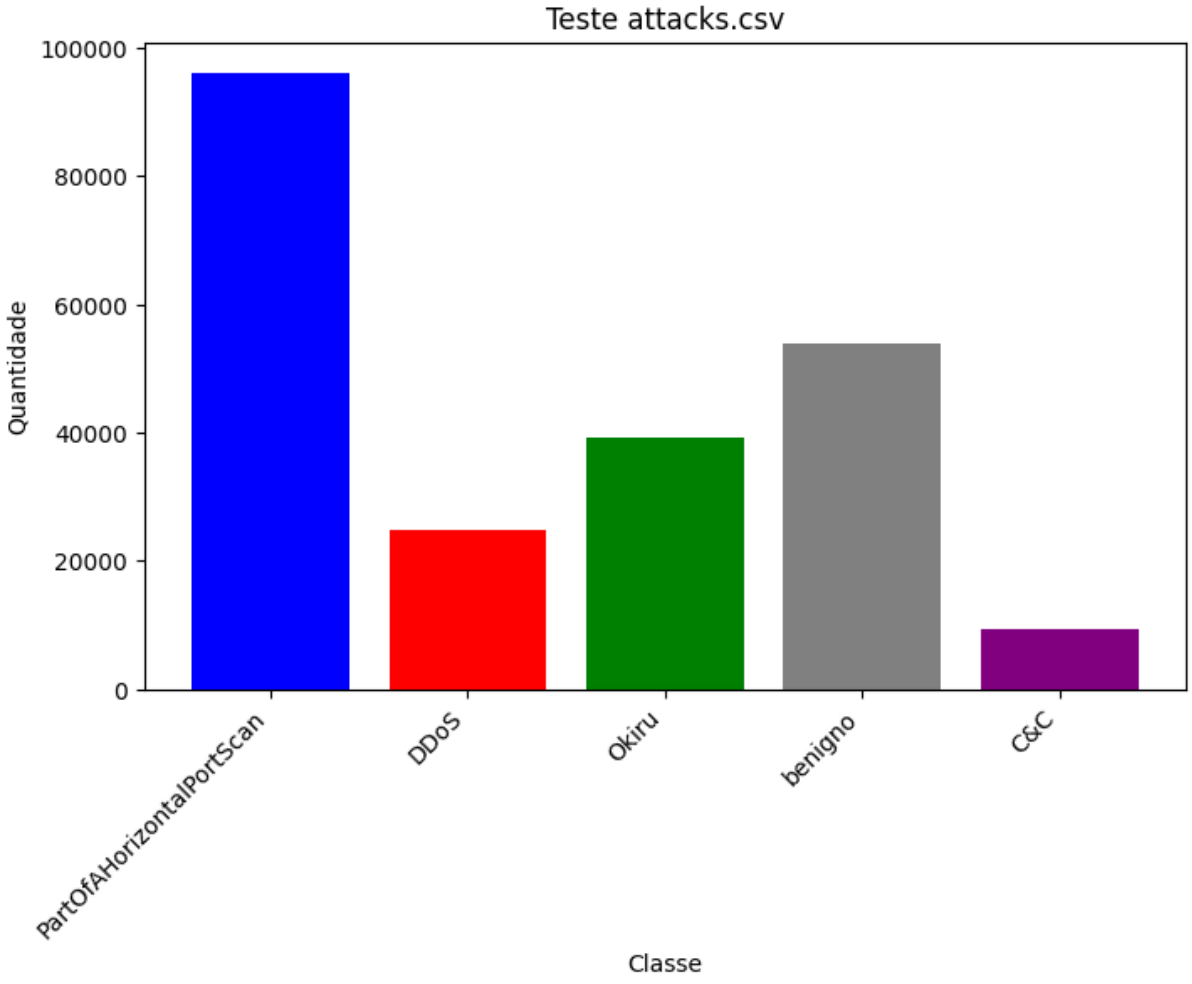
Fonte: elaborada pelo autor.

As Figuras 16, 17, 18 e 19 resumem as classificações feitas pelo modelo *tinyml* para cada arquivo de teste. Vemos que o modelo classifica corretamente, mas apresenta erros. Por exemplo, o arquivo *benign.csv* apresenta apenas amostras de tráfego benigno, mas a Figura 17 mostra que outras classes foram detectadas.

5.4 Discussão

Os resultados obtidos demonstram a efetividade da privacidade diferencial na mitigação de ataques de inferência de associação em modelos de aprendizado profundo aplicados à detecção de ataques na internet das coisas. No entanto, como esperado, essa proteção vem acompanhada de uma perda de desempenho do modelo classificador. Observou-se que, à medida que o parâmetro de privacidade ε diminuiu, a acurácia do modelo também sofreu um impacto significativo, caindo de 99% no modelo *baseline* para valores próximos a 64% nos cenários mais restritivos. Apesar dessa degradação, o ataque de inferência de associação também teve sua eficácia reduzida, demonstrando que a adição de ruído ao treinamento contribui para evitar vazamento de informações sensíveis utilizadas na construção do modelo. Além disso, os experi-

Figura 16 – Classificação do modelo *tinyml* para arquivo de teste *attacks.csv*

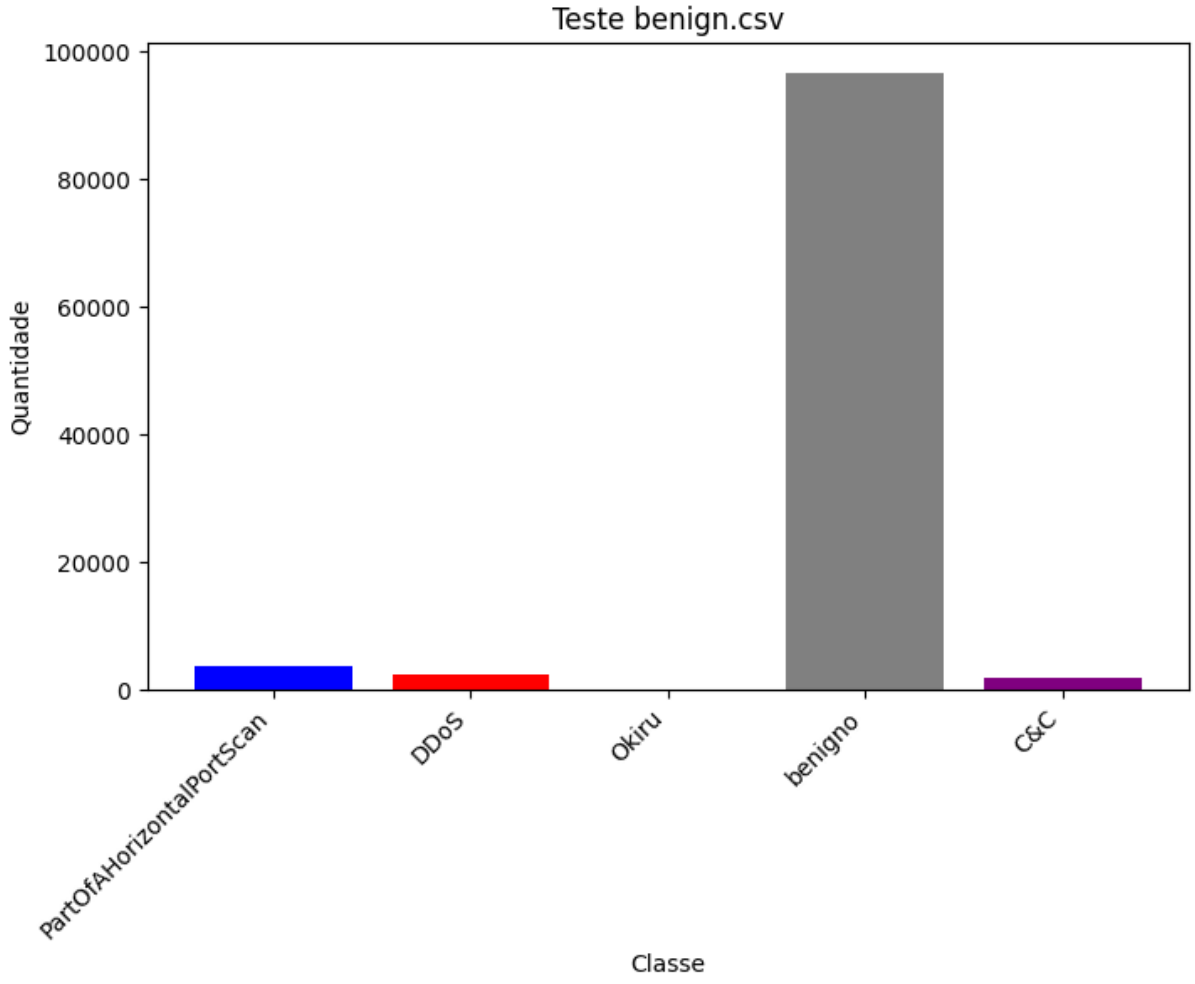


Fonte: Elaborada pelo autor.

mentos mostraram que normas de corte mais elevadas resultaram em um melhor equilíbrio entre privacidade e desempenho, sugerindo que um ajuste criterioso desse parâmetro pode permitir um melhor *trade-off*.

A otimização do modelo com *tinyml* mostrou-se um fator crucial para viabilizar sua implementação em dispositivos de borda, reduzindo significativamente os requisitos computacionais sem comprometer severamente o desempenho. A conversão e aplicação de técnicas de quantização resultaram em uma redução expressiva no consumo de memória, passando de 8,35 MB para apenas 0,05 MB. Essa otimização permitiu uma execução mais eficiente, tornando o modelo adequado para ambientes com recursos limitados, como dispositivos IoT. No entanto, a aplicação da quantização de 8 *bits* trouxe um impacto perceptível na precisão do modelo, sugerindo que, embora a redução no consumo de recursos seja vantajosa, existe um ponto de equilíbrio a ser considerado. A perda de desempenho decorrente de quantizações mais agressivas pode comprometer a confiabilidade do classificador, tornando essencial uma análise criteriosa

Figura 17 – Classificação do modelo *tinyml* para arquivo de teste *benign.csv*

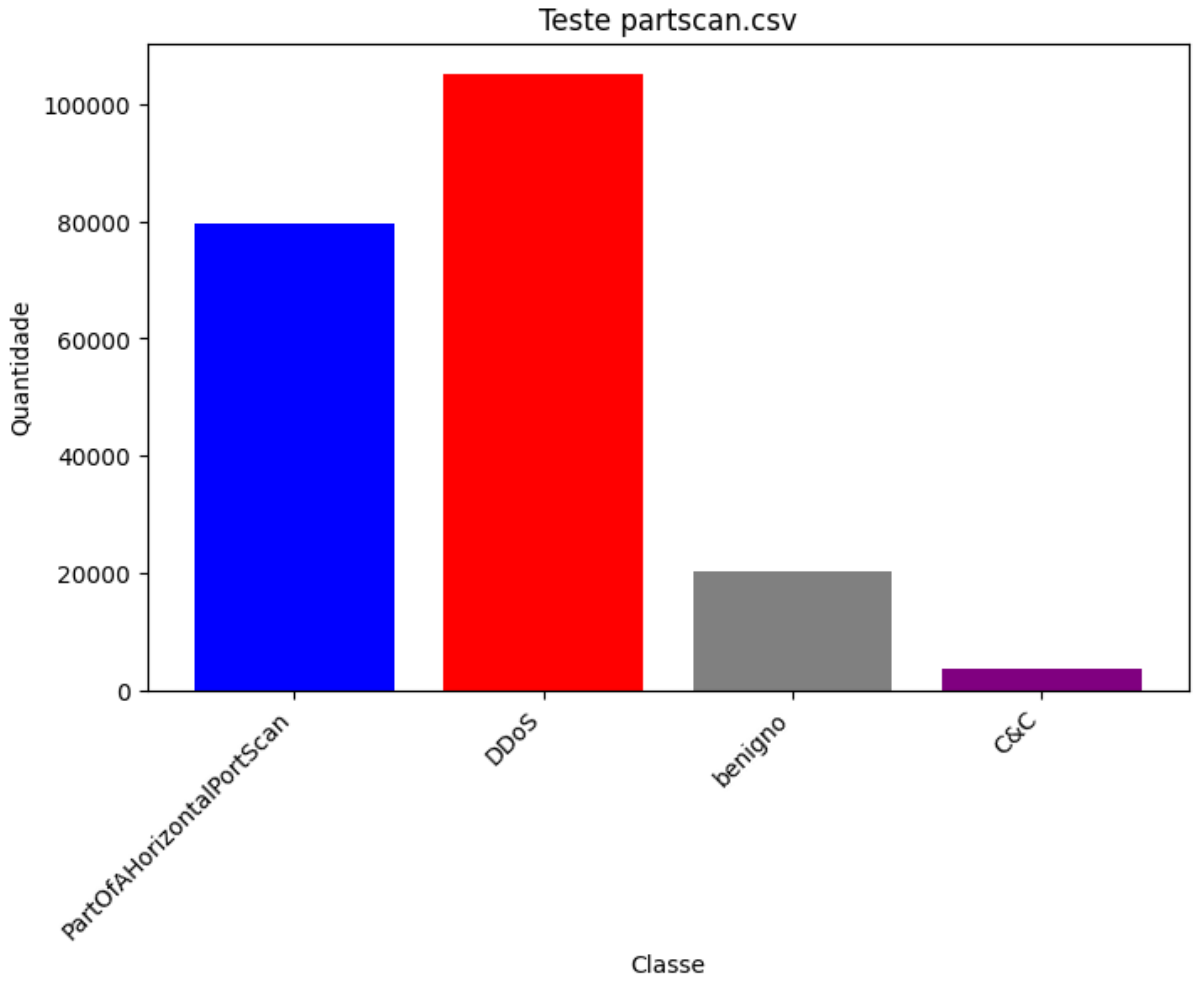


Fonte: Elaborada pelo autor.

para determinar o nível de otimização que mantém a acurácia dentro de limites aceitáveis.

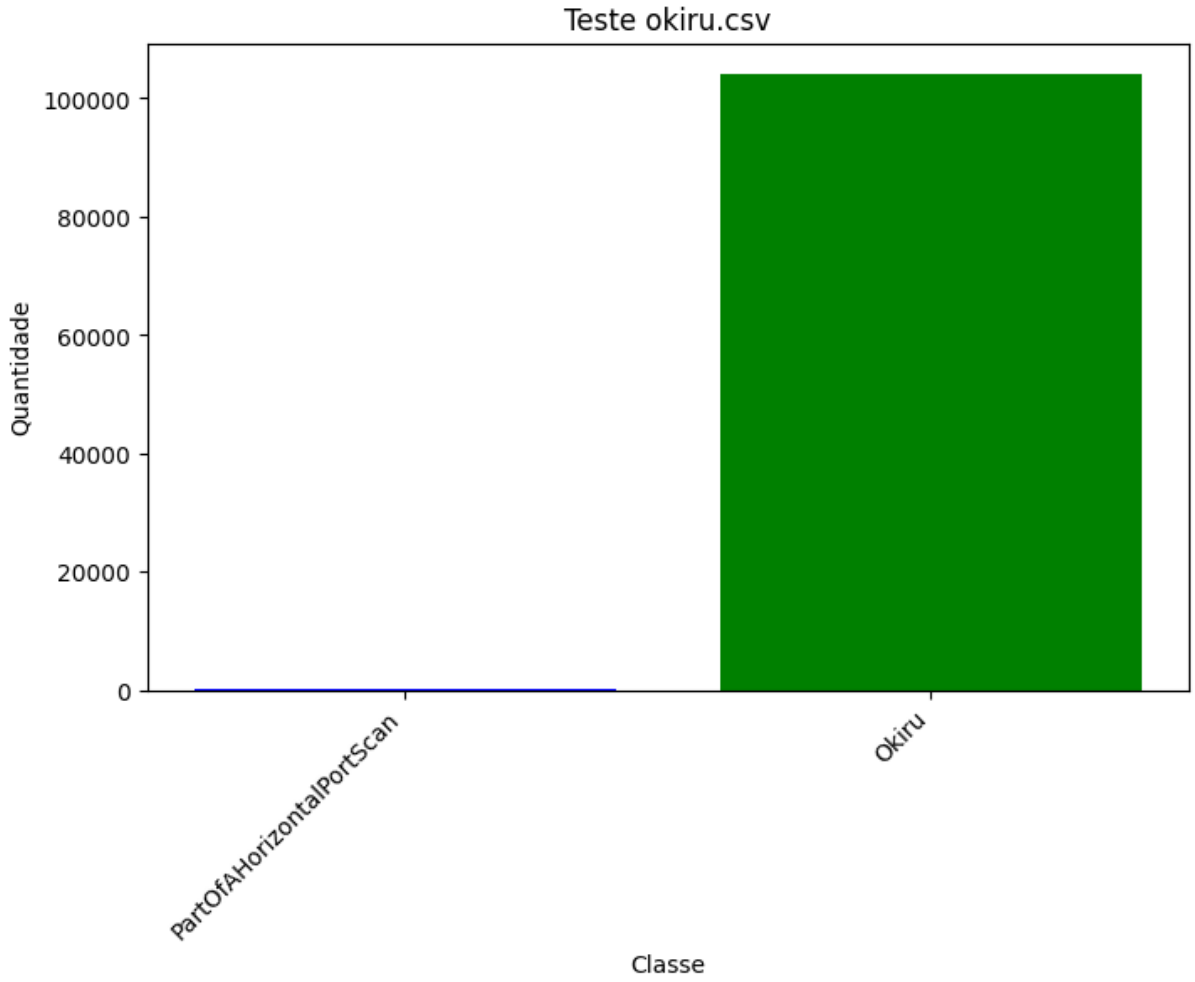
Os resultados mostram que combinar privacidade diferencial com a otimização para *tinyml* pode ser uma boa estratégia para detectar ataques em redes IoT, mas essa abordagem precisa ser bem planejada para manter um equilíbrio entre segurança, desempenho e eficiência computacional. A privacidade diferencial ajuda a dificultar ataques que tentam descobrir informações usadas no treinamento do modelo, mas, ao mesmo tempo, pode reduzir a capacidade do sistema de classificar corretamente os dados. Além disso, a otimização do modelo para rodar em dispositivos com poucos recursos também pode afetar sua precisão. Por isso, é importante escolher com cuidado os parâmetros de privacidade, como o nível de ruído adicionado ao treinamento, e as técnicas de otimização, como a redução do tamanho do modelo.

Figura 18 – Classificação do modelo *tinyml* para arquivo de teste *partscan.csv*



Fonte: Elaborada pelo autor.

Figura 19 – Classificação do modelo *tinyml* para arquivo de teste *okiru.csv*



Fonte: Elaborada pelo autor.

6 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho investigou a aplicação de Privacidade Diferencial na detecção de ataques em redes IoT, avaliando o impacto do uso do otimizador DP-ADAM em um modelo *feedforward* otimizado para *tinyml*. Os resultados demonstraram que a privacidade diferencial reduz significativamente a eficácia do ataque de inferência de associação, tornando o modelo menos suscetível à exposição de dados do treinamento. No entanto, essa proteção ocorre às custas da acurácia do modelo, evidenciando um *trade-off* entre privacidade e utilidade.

Observou-se que normas de corte menores impõem restrições mais rígidas aos gradientes do modelo, resultando em maior proteção contra ataques de inferência, mas também em uma perda expressiva de desempenho na classificação. Já normas de corte maiores oferecem um melhor equilíbrio entre utilidade e privacidade, permitindo que o modelo mantenha um desempenho mais satisfatório sem comprometer excessivamente a proteção contra ataques.

Além disso, a otimização com *tinyml* mostrou-se eficaz na redução do tempo de inferência e no consumo de memória, tornando o modelo viável para execução em dispositivos de borda. A conversão do modelo para *TensorFlow Lite* permitiu uma execução até 10 vezes mais rápida, enquanto a quantização de 8 bits reduziu significativamente o consumo computacional, sem comprometer a precisão do modelo. Esses resultados destacam o potencial do *tinyml* para aplicações de segurança em IoT, possibilitando a implementação de modelos eficientes e diferencialmente privados em dispositivos com recursos limitados. Com isso, torna-se viável o uso de aprendizado profundo em ambientes restritos, sem depender de infraestrutura robusta de processamento.

REFERÊNCIAS

ABADADE, Y.; TEMOUDEN, A.; BAMOUMEN, H.; BENAMAR, N.; CHTOUKI, Y.; HAFID, A. S. A comprehensive survey on tinyml. **IEEE Access**, IEEE, 2023. Disponível em: https://ieeexplore.ieee.org/document/10177729. Acesso em: 12 jun. 2024.

ABADI, M.; CHU, A.; GOODFELLOW, I.; MCMAHAN, H. B.; MIRONOV, I.; TALWAR, K.; ZHANG, L. Deep learning with differential privacy. In: **Proceedings of the 2016 ACM SIGSAC conference on computer and communications security**. [*S. n.*], 2016. p. 308–318. Disponível em: https://arxiv.org/abs/1607.00133. Acesso em: 02 mai. 2024.

AHMAD, Z.; KHAN, A. S.; SHIANG, C. W.; ABDULLAH, J.; AHMAD, F. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. **Transactions on Emerging Telecommunications Technologies**, Wiley Online Library, v. 32, n. 1, p. e4150, 2021. Disponível em: https://onlinelibrary.wiley.com/doi/full/10.1002/ett.4150. Acesso em: 14 jul. 2024.

AI, T. **Adversarial Robustness Toolbox repository**. 2024. Https://github.com/Trusted-AI/adversarial-robustness-toolbox. Disponível em: https://github.com/Trusted-AI/adversarial-robustness-toolbox. Acesso em: 21 dez. 2024.

ALFANDI, O.; KHANJI, S.; AHMAD, L.; KHATTAK, A. A survey on boosting iot security and privacy through blockchain: Exploration, requirements, and open issues. **Cluster Computing**, Springer, v. 24, n. 1, p. 37–55, 2021. Disponível em: https://link.springer.com/article/10.1007/s10586-020-03137-8. Acesso em: 13 abr. 2024.

BANBURY, C. R.; REDDI, V. J.; LAM, M.; FU, W.; FAZEL, A.; HOLLEMAN, J.; HUANG, X.; HURTADO, R.; KANTER, D.; LOKHMOTOV, A. *et al.* Benchmarking tinyml systems: Challenges and direction. **arXiv preprint arXiv:2003.04821**, 2020. Disponível em: https://arxiv.org/abs/2003.04821.

BEZERRA, E. Introdução à aprendizagem profunda. **Artigo–31º Simpósio Brasileiro de Banco de Dados–SBBD2016–Salvador**, 2016. Disponível em: https://www.researchgate.net/publication/309321510_Introducao_a_Aprendizagem_Profunda. Acesso em: 15 ago. 2024.

BOULEMTAFES, A.; DERHAB, A.; CHALLAL, Y. A review of privacy-preserving techniques for deep learning. **Neurocomputing**, Elsevier, v. 384, p. 21–45, 2020. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S0925231219316431. Acesso em: 18 ago. 2024.

CAO, K.; LIU, Y.; MENG, G.; SUN, Q. An overview on edge computing research. **IEEE access**, IEEE, v. 8, p. 85714–85728, 2020. Disponível em: https://ieeexplore.ieee.org/document/9083958. Acesso em: 21 abr. 2024.

CARNAZ, G.; NOGUEIRA, V. B. An overview of iot and healthcare. Escola de Ciências e Tecnologia da Universidade de Évora, 2016. Disponível em: https://dspace.uevora.pt/rdpc/handle/10174/19998.

CHAWLA, N. V.; BOWYER, K. W.; HALL, L. O.; KEGELMEYER, W. P. Smote: synthetic minority over-sampling technique. **Journal of artificial intelligence research**, v. 16, p. 321–357, 2002. Disponível em: https://arxiv.org/abs/1106.1813. Acesso em: 11 dez. 2024.

CHEN, W.; JEONG, S.; JUNG, H. Wifi-based home iot communication system. **Journal of information and communication convergence engineering**, The Korea Institute of Information and Commucation Engineering, v. 18, n. 1, p. 8–15, 2020. Disponível em: https://www.koreascience.or.kr/article/JAKO202010163509620.pdf.

CHOLLET, F. *et al.* **Keras**. 2015. https://keras.io. Disponível em: https://keras.io. Acesso em: 04 ago. 2024.

CHUA, L.; GHAZI, B.; KAMATH, P.; KUMAR, R.; MANURANGSI, P.; SINHA, A.; ZHANG, C. How private is dp-sgd? **arXiv preprint arXiv:2403.17673**, 2024. Disponível em: https://arxiv.org/abs/2403.17673. Acesso em: 26 set. 2024.

DAMGHANI, H.; DAMGHANI, L.; HOSSEINIAN, H.; SHARIFI, R. Classification of attacks on iot. In: **4th international conference on combinatorics, cryptography, computer science and computation**. [*S. n.*], 2019. p. 245–255. Disponível em: https://www.academia.edu/44836149/Classification_of_Attacks_on_IoT. Acesso em: 9 dez. 2024.

DUTTA, A.; KANT, S. Implementation of cyber threat intelligence platform on internet of things (iot) using tinyml approach for deceiving cyber invasion. In: IEEE. **2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)**. 2021. p. 1–6. Disponível em: https://ieeexplore.ieee.org/document/9590959.

DUTTA, L.; BHARALI, S. Tinyml meets iot: A comprehensive survey. **Internet of Things**, Elsevier, v. 16, p. 100461, 2021. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S2542660521001025.

DWORK, C. Differential privacy. In: SPRINGER. **International colloquium on automata, languages, and programming**. 2006. p. 1–12. Disponível em: https://link.springer.com/chapter/10.1007/11787006_1. Acesso em: 4 jul. 2024.

GARCIA AGUSTIN PARMISANO, M. J. E. S. **IoT-23: A labeled dataset with malicious and benign IoT network traffic (Version 1.0.0)**. 2020. Http://doi.org/10.5281/zenodo.4743746. Disponível em: http://doi.org/10.5281/zenodo.4743746. Acesso em: 20 jun. 2024.

GYAMFI, E.; JURCUT, A. Intrusion detection in internet of things systems: a review on design approaches leveraging multi-access edge computing, machine learning, and datasets. **Sensors**, MDPI, v. 22, n. 10, p. 3744, 2022. Disponível em: https://www.mdpi.com/1424-8220/22/10/3744. Acesso em: 02 abr. 2024.

HADDADI, F.; KHANCHI, S.; SHETABI, M.; DERHAMI, V. Intrusion detection and attack classification using feed-forward neural network. In: IEEE. **2010 Second international conference on computer and network technology**. 2010. p. 262–266. Disponível em: https://ieeexplore.ieee.org/document/5474495. Acesso em: 12 ago. 2024.

HU, H.; SALCIC, Z.; SUN, L.; DOBBIE, G.; YU, P. S.; ZHANG, X. Membership inference attacks on machine learning: A survey. **ACM Computing Surveys (CSUR)**, ACM New York, NY, v. 54, n. 11s, p. 1–37, 2022. Disponível em: https://arxiv.org/abs/2103.07853. Acesso em: 14 jan. 2025.

HUBARA, I.; NAHSHAN, Y.; HANANI, Y.; BANNER, R.; SOUDRY, D. Accurate post training quantization with small calibration sets. In: PMLR. **International**

Conference on Machine Learning. 2021. p. 4466–4475. Disponível em: https://proceedings.mlr.press/v139/hubara21a/hubara21a.pdf. Acesso em: 26 fev. 2025.

HYMEL, S.; BANBURY, C.; SITUNAYAKE, D.; ELIUM, A.; WARD, C.; KELCEY, M.; BAAIJENS, M.; MAJCHRZYCKI, M.; PLUNKETT, J.; TISCHLER, D. *et al.* Edge impulse: An mlops platform for tiny machine learning. **arXiv preprint arXiv:2212.03332**, 2022. Disponível em: https://arxiv.org/abs/2212.03332. Acesso em: 10 nov. 2024.

IBM. **O que é o gradiente descendente?** 2023. Https:www.ibm.combr-pttopicsgradient-descent. Disponível em: https://www.ibm.com/br-pt/think/topics/gradient-descent#:~:text=O%20gradiente%20descendente%20%C3%A9%20um,resultados%20previstos%20e%20os%20reais. Acesso em: 05 abr. 2024.

JYOTHSNA, V.; PRASAD, R.; PRASAD, K. M. A review of anomaly based intrusion detection systems. **International Journal of Computer Applications**, Citeseer, v. 28, n. 7, p. 26–35, 2011. Disponível em: https://www.researchgate.net/publication/271155860_A_Review_of_Anomaly_based_Intrusion_Detection_Systems. Acesso em: 12 set. 2024.

KARATAS, G.; DEMIR, O.; SAHINGOZ, O. K. Deep learning in intrusion detection systems. In: IEEE. **2018 international congress on big data, deep learning and fighting cyber terrorism (IBIGDELFT)**. 2018. p. 113–116. Disponível em: https://ieeexplore.ieee.org/document/8625278/. Acesso em: 8 dez. 2024.

KHAN, W. Z.; AHMED, E.; HAKAK, S.; YAQOOB, I.; AHMED, A. Edge computing: A survey. **Future Generation Computer Systems**, Elsevier, v. 97, p. 219–235, 2019. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S0167739X18319903. Acesso em: 21 abr. 2024.

KIM, S.; PARK, G.; YI, Y. Performance evaluation of int8 quantized inference on mobile gpus. **IEEE Access**, IEEE, v. 9, p. 164245–164255, 2021. Disponível em: https://ieeexplore.ieee.org/document/9638444. Acesso em: 10 jan. 2025.

KRISHNAMOORTHI, R. Quantizing deep convolutional networks for efficient inference: A whitepaper. arxiv 2018. **arXiv preprint arXiv:1806.08342**, 2018. Disponível em: https://arxiv.org/abs/1806.08342. Acesso em: 12 jun. 2024.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **nature**, Nature Publishing Group UK London, v. 521, n. 7553, p. 436–444, 2015. Disponível em: https://www.nature.com/articles/nature14539. Acesso em: 27 abr. 2024.

MACHOOKA, D.; YUAN, X.; ROY, K.; CHEN, G. Differential privacy with dp-sgd and pate for intrusion detection: A comparative study. In: IEEE. **2025 IEEE 4th International Conference on AI in Cybersecurity (ICAIC)**. 2025. p. 1–7. Disponível em: https://www.researchgate.net/publication/388722914_Differential_Privacy_with_DP-SGD_and_PATE_for_Intrusion_Detection_A_Comparative_Study. Acesso em: 8 nov. 2024.

MUKHOPADHYAY, S. C.; SURYADEVARA, N. K. **Internet of things: Challenges and opportunities**. [*S. l.*]: Springer, 2014. Https://link.springer.com/chapter/10.1007/978-3-319-04223-7$_1$.*Acessoem* : 22*mai*.2024.

PEDRO, R. R. F. Privacidade diferencial para proteger dados sobre utilização de bicicletas compartilhadas. 2022. Disponível em: https://sites.uel.br/dc/wp-content/uploads/2022/09/TCC_RENAN_RICOLDI_FROIS_PEDRO.pdf. Acesso em: 15 jul. 2024.

PONTI, M. A.; COSTA, G. B. P. D. Como funciona o deep learning. **arXiv preprint arXiv:1806.07908**, 2018. Disponível em: https://arxiv.org/abs/1806.07908. Acesso em: 02 mai. 2024.

POUYANFAR, S.; SADIQ, S.; YAN, Y.; TIAN, H.; TAO, Y.; REYES, M. P.; SHYU, M.-L.; CHEN, S.-C.; IYENGAR, S. S. A survey on deep learning: Algorithms, techniques, and applications. **ACM computing surveys (CSUR)**, ACM New York, NY, USA, v. 51, n. 5, p. 1–36, 2018. Disponível em: https://dl.acm.org/doi/10.1145/3234150. Acesso em: 22 abr. 2024.

PRIVACY tensorflow. **tensorflow privacy - DPKerasAdamOptimizer**. 2024. Disponível em: https://www.tensorflow.org/responsible_ai/privacy/api_docs/python/tf_privacy/DPKerasAdamOptimizer. Acesso em: 11 nov. 2024.

PUSTOZEROVA, A.; BAUMBACH, J.; MAYER, R. Differentially private federated learning: Privacy and utility analysis of output perturbation and dp-sgd. In: IEEE. **2023 IEEE International Conference on Big Data (BigData)**. 2023. p. 5549–5558. Disponível em: https://ieeexplore.ieee.org/document/10386466/. Acesso em: 22 dez. 2024.

PYPI. **tflite-runtime 2.14.0 documentation**. 2023. Https://pypi.org/project/tflite-runtime/. Disponível em: https://pypi.org/project/tflite-runtime/. Acesso em: 12 out. 2024.

QI, X.; WANG, T.; LIU, J. Comparison of support vector machine and softmax classifiers in computer vision. In: IEEE. **2017 Second International Conference on Mechanical, Control and Computer Engineering (ICMCCE)**. 2017. p. 151–155. Disponível em: https://ieeexplore.ieee.org/document/8269910. Acesso em: 19 ago. 2024.

RASPBERRYPI. **Raspberry Pi OS**. 2024. Https:www.raspberrypi.comsoftware. Disponível em: https://www.raspberrypi.com/software/. Acesso em: 16 jun. 2024.

REIS, C. H. Otimização de hiperparâmetros em redes neurais profundas. **Minas Gerais**, 2021. Disponível em: https://carlos-henreis.github.io/files/Monografia_TFG.pdf. Acesso em: 24 jan. 2025.

REN, H.; LI, X.; ANICIC, D.; RUNKLER, T. A. Tinymetafed: Efficient federated meta-learning for tinyml. **arXiv preprint arXiv:2307.06822**, 2023. Disponível em: https://arxiv.org/abs/2307.06822. Acesso em: 16 jul. 2024.

RIMER, M.; MARTINEZ, T. Softprop: softmax neural network backpropagation learning. In: IEEE. **2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541)**. 2004. v. 2, p. 979–983. Disponível em: https://ieeexplore.ieee.org/document/1380066. Acesso em: 15 jan. 2025.

RODRIGUES, P. S. A. M.; ABREU, R. O. Ameaças e vulnerabilidades em dispositivos iot. 004, 2021. Disponível em: https://ric.cps.sp.gov.br/handle/123456789/12806.

SHI, W.; DUSTDAR, S. The promise of edge computing. **Computer**, IEEE, v. 49, n. 5, p. 78–81, 2016. Disponível em: https://ieeexplore.ieee.org/document/7469991. Acesso em: 19 mai. 2024.

SIACHOS, I.; KALTAKIS, K.; PAPACHRISTOPOULOU, K.; GIANNOULAKIS, I.; KAFETZAKIS, E. Comparison of machine learning algorithms trained under differential privacy for intrusion detection systems. In: IEEE. **2023 IEEE International Conference on Cyber Security and Resilience (CSR)**. 2023. p. 654–658. Disponível em: https://ieeexplore.ieee.org/document/10225010/. Acesso em: 13 out. 2024.

SIANG, Y. Y.; AHAMD, M. R.; ABIDIN, M. S. Z. Anomaly detection based on tiny machine learning: A review. **Open International Journal of Informatics**, v. 9, n. Special Issue 2, p. 67–78, 2021. Disponível em: https://oiji.utm.my/index.php/oiji/article/view/148.

TANG, Q.; SHPILEVSKIY, F.; LÉCUYER, M. Dp-adambc: Your dp-adam is actually dp-sgd (unless you apply bias correction). In: **Proceedings of the AAAI Conference on Artificial Intelligence**. [*S. n.*], 2024. v. 38, n. 14, p. 15276–15283. Disponível em: https://arxiv.org/abs/2312.14334. Acesso em: 24 nov. 2024.

TENSORFLOW. **Tensorflow Lite**. 2021. Https://www.tensorflow.org/lite/guide?hl=pt-br. Disponível em: https://www.tensorflow.org/lite/guide?hl=pt-br. Acesso em: 20 dez. 2024.

TENSORFLOW. **Quantização pós-treinamento**. 2024. Https:www.tensorflow.orgliteperformancepost_training_quantization?hl=pt-br. Disponível em: https://ai.google.dev/edge/litert/models/post_training_quantization?hl=pt-br. Acesso em: 14 jun. 2024.

TENSORFLOW. **Tensorflow Lite Interpreter**. 2024. Https://www.tensorflow.org/api$_{d}ocs/python/tf/lite/I$

.

TENSORFLOW. **Tensorflow Privacy**. 2024. Disponível em: https://github.com/tensorflow/privacy. Acesso em: 11 mai. 2024.

TOOLBOX, A. R. **Membership inference attacks**. 2018. Https://adversarial-robustness-toolbox.readthedocs.io/en/latest/modules/attacks/inference/membership_inference.html. Disponível em: https://adversarial-robustness-toolbox.readthedocs.io/en/latest/modules/attacks/inference/membership_inference.html. Acesso em: 14 jan. 2025.

TSIMENIDIS, S.; LAGKAS, T.; RANTOS, K. Deep learning in iot intrusion detection. **Journal of network and systems management**, Springer, v. 30, n. 1, p. 8, 2022. Disponível em: https://www.researchgate.net/publication/355162850_Deep_Learning_in_IoT_Intrusion_Detection.

VIDAL, I. d. C. Protecting: garantindo a privacidade de dados gerados em casas inteligentes localmente na borda da rede. 2020. Disponível em: https://repositorio.ufc.br/handle/riufc/56749. Acesso em: 25 set. 2024.

VILLEGAS-CH, W.; GUTIERREZ, R.; NAVARRO, A. M.; MERA-NAVARRETE, A. Optimizing federated learning on tinyml devices for privacy protection and energy efficiency in iot networks. **IEEE Access**, IEEE, 2024. Disponível em: https://ieeexplore.ieee.org/document/10758420. Acesso em: 24 fev. 2025.

WANG, Q.; ZHU, X.; NI, Y.; GU, L.; ZHU, H. Blockchain for the iot and industrial iot: A review. **Internet of Things**, Elsevier, v. 10, p. 100081, 2020. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S254266051930085X.

WANG, Y.; WANG, Q.; ZHAO, L.; WANG, C. Differential privacy in deep learning: Privacy and beyond. **Future Generation Computer Systems**, Elsevier, v. 148, p. 408–424, 2023. Disponível em: https://github.com/cloudera/CML_AMP_Anomaly_Detection.

WU, D.; QI, S.; QI, Y.; LI, Q.; CAI, B.; GUO, Q.; CHENG, J. Understanding and defending against white-box membership inference attack in deep learning. **Knowledge-Based Systems**, Elsevier, v. 259, p. 110014, 2023. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S0950705122011078. Acesso em: 21 dez. 2024.

ZHANG, Z.; YAN, C.; MALIN, B. A. Membership inference attacks against synthetic health data. **Journal of biomedical informatics**, Elsevier, v. 125, p. 103977, 2022. Disponível em: https://www.sciencedirect.com/science/article/pii/S1532046421003063. Acesso em: 26 jan. 2025.