UNIVERSIDADE FEDERAL DE SANTA CATARINA
CURSO DE GRADUAÇÃO EM CIÊNCIAS DA COMPUTAÇÃO

Abordagem de Detecção e Identificação de Intrusão baseada em Redes Neurais no contexto de Internet das Coisas

JOÃO VITOR CARDOSO

FLORIANÓPOLIS
2020/2

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CURSO DE GRADUAÇÃO EM CIÊNCIAS DA COMPUTAÇÃO

Abordagem de Detecção e Identificação de Intrusão baseada em Redes Neurais no contexto de Internet das Coisas

JOÃO VITOR CARDOSO

FLORIANÓPOLIS
2020/2

Abordagem de Detecção e Identificação de Intrusão baseada em Redes Neurais no contexto de Internet das Coisas

Trabalho de Conclusão de Curso de Graduação em Ciências da Computação, do Departamento de Informática e Estatística, do Centro Tecnológico da Universidade Federal de Santa Catarina, requisito parcial à obtenção do título de Bacharel em Ciências da Computação.

Orientador: Carlos Becker Westphall, Prof. Dr.

Coorientador: Cristiano Antonio de Souza, Me.

FLORIANÓPOLIS

2020/2

Abordagem de Detecção e Identificação de Intrusão baseada em Redes Neurais no contexto de Internet das Coisas

JOÃO VITOR CARDOSO

Trabalho de Conclusão de Curso de Graduação em Ciências da Computação, do Departamento de Informática e Estatística, do Centro Tecnológico da Universidade Federal de Santa Catarina, requisito parcial à obtenção do título de Bacharel em Ciências da Computação.

Aprovado em:

Por:

Prof. Dr. Carlos Becker Westphall
Orientador

Me. Cristiano Antonio de Souza
Coorientador

Prof. Dra. Jerusa Marchi
Membro da banca

Prof. Dra. Carla Merkle Westphall
Membro da banca

AGRADECIMENTOS

Primeiramente agradeço a minha família que sempre me apoiou nesta caminhada.

Aos meus amigos que me acompanharam durante o curso, as aventuras vividas e por estarem presente nos momentos bons e ruins.

Aos meus professores, por tudo o que me ensinaram durante esse tempo.

A meu orientador, Carlos Becker Westphall, e coorientador, Cristiano Antonio de Souza, pela ajuda e apoio no desenvolvimento deste trabalho.

"Bem-aventurados os que têm fome e sede de justiça, porque eles serão fartos"

— Matheus 5:6

RESUMO

Internet das Coisas (IoT) é um paradigma presente em aplicações de diversas áreas, tais como área médica, agricultura, automação residencial, dentre outras. Atualmente dispositivos eletrônicos, para uso doméstico, estão se tornando cada vez mais populares. Devido às restrições de recursos que estes dispositivos possuem, eles podem apresentar vulnerabilidades relacionadas à segurança, que podem ser exploradas por indivíduos mal intencionados. Atualmente, trabalhos utilizando redes neurais para solucionar problemas de classificação estão obtendo resultados positivos, inclusive no contexto de detecção de intrusão. Neste trabalho será investigada a estratégia de decomposição de problemas chamada *One vs All*. Esta estratégia explora a alta taxa de precisão que os classificadores binários possuem e os utiliza para resolver problemas multiclasse. Este trabalho propõe usar esta estratégia em conjunto com modelos de redes neurais *feedfoward* para detecção de intrusão em sistemas IoT. Além disso, propõem comparar o modelo criado com outras técnicas clássicas.

Palavras-chave: Redes Neurais Profundas; Sistema de Detecção de Intrusão; Internet das Coisas; Computação em Nevoeiro

ABSTRACT

Internet of Things (IoT) is a paradigm present in applications from different areas, such as medical, agriculture, home automation, among others. Today, electronic devices for home use are becoming increasingly popular. Due to the resource restrictions that these devices have, they can present security-related vulnerabilities, which can be exploited by malicious individuals. Currently, the use of neural networks to solve classification problems is getting very positive results, including in the context of intrusion detection. In this work the problem decomposition strategy called One vs All will be investigated. This strategy explores the high rate of precision that binary classifiers have and uses them to solve multiclass problems. This work proposes to use this strategy in conjunction with models of deep neural networks for intrusion detection in IoT systems. In addition, they propose to compare the model created with other classic techniques.

Key-words: Deep Neural Networks, Intrusion Detection System, Internet of Things, Fog Computing

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DDoS	*Distributed Denial of Service*
DoS	*Denial Of Service*
DT	*Decision Tree*
Fog	*Fog Computing*
GRU	*Gated Recurrent Unit*
HIDS	*Host-based Intrusion Detection System*
IaaS	*Infrastructure as a service*
IDS	*Intrusion Detection System*
IEEE	*Institute of Electrical and Electronics Engineers*
IoT	*Internet of Things*
IPS	*Intrusion Prevention System*
kNN	*k-Nearest Neighbors*
LSM	*Long-Short-Memory*
MDPI	*Multidisciplinary Digital Publishing Institute*
MLP	*Multi Layer Perceptron*
NIDS	*Network-based Intrusion Detection System*
PaaS	*Platform as a service*
RNA	Redes Neurais Artificiais
RNN	Rede Neural Recorrente
ROC	*Operating Characteristic Curve*
SaaS	*Software as a service*
SFS	*Sequence Forward Selection*
SFSDT	*Sequence Forward Selection Decision Tree*

SQL	*Structured Query Language*
SVM	*Support Vector Machine*
TCP/IP	*Transmission Control Protocol/Internet Protocol*
TNR	*True Negative Rate*
UDP	*User Datagram Protocol*

SUMÁRIO

1 INTRODUÇÃO

Atualmente, existem diversos dispositivos eletrônicos e eletrodomésticos que podem ser conectados à Internet, disponibilizando dados e serviços remotos para os usuários (Vikram *et al.*, 2017). Por exemplo, existem tomadas e lâmpadas que podem ser ligadas à rede sem fio, sendo possível ligá-las e desligá-las remotamente, agendando horário para tais funcionalidades. Esses dispositivos podem ser utilizados para projetar ambientes IoT, como automação residencial.

Ambientes IoT são compostos por dispositivos com recursos restritos, baixa capacidade de processamento e memória limitada. Esses dispositivos normalmente possuem sensores que coletam informações do ambiente, gerando grande quantidade de dados. Para realizar o armazenamento, processamento e análise desses dados são utilizados sistemas em nuvem. O grande tráfego gerado por esses dispositivos e a latência causada pela distância entre o servidor em Nuvem e os dispositivos IoT é um grande problema para esses sistemas. Por isso surgiu o paradigma de *Fog Computing* que adiciona um nó intermediário fazendo um pré-processamento antes de enviar os dados para a nuvem (Iorga *et al.*, 2018).

Conforme supracitado, os dispositivos IoT possuem baixa capacidade de processamento e isso dificulta a adição de mecanismo de segurança (Frustaci *et al.*, 2017). Casos de violação de segurança são comuns em sistemas IoT, pois muitas vezes a segurança não recebe a devida atenção no projeto destes sistemas.

Uma abordagem para prover segurança a estes sistemas é tentar identificar ataques utilizando mecanismos como Sistema de Detecção de Intrusão baseado em rede. Este tipo de ferramenta analisa o tráfego, sem interferir na comunicação entre os dispositivos, para descobrir se um ataque está sendo realizado (Soe *et al.*, 2019; Garcia-Teodoro *et al.*, 2009).

Além de detectar a ocorrência do ataque é importante também identificar o tipo de ataque que está ocorrendo para auxiliar na tomada de decisão do responsável da rede. Segundo os autores (Hughes; McLaughlin; Sezer, 2020) Sistemas de Resposta de Intrusão, que visam responder a ataques automaticamente assim que são alertados, necessitam de conhecimentos específicos para aplicar contramedidas precisas e eficazes.

1.1 MOTIVAÇÃO

A Internet das Coisas está cada vez mais presente no dia a dia das pessoas por meio de aparelhos usados no seu cotidiano, mas ainda possui muitas vulnerabilidades. Em 2016, muitos dispositivos IoT foram invadidos, através do *malware Mirai*, e utilizados para fazer ataques de negação de serviço, prejudicando algumas funcionalidades de serviços, como *Twitter*, *Spotify*, tornando-os indisponíveis. A principal motivação deste trabalho é melhorar a classificação do comportamento intrusivo em sistemas que estão cada vez mais próximo das pessoas.

1.2 OBJETIVOS

Nesta seção são apresentados os objetivos deste trabalho.

1.2.1 Objetivos gerais

Este trabalho tem como objetivo geral investigar o uso da estratégia de decomposição de classe *One Vs All* com o modelo de redes neurais *perceptron* multicamadas para a detecção de intrusão no contexto de Internet das Coisas.

1.2.2 Objetivos específicos

Como objetivos específicos, pode-se listar:

1. Propor um método de detecção e identificação de intrusão baseado em detecção multiclasse utilizando a estratégia de decomposição de classes *One Vs All* em conjunto com um modelo de rede neural feedforward com multiplas camadas;
2. Avaliar a solução criada utilizando métricas de desempenho de classficação;
3. Comparar o modelo com os métodos clássicos de *machine learning* e abordagens de trabalhos correlatos.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 INTERNET DAS COISAS

Internet das Coisas consiste na integração de um conjunto de objetos físicos, como sensores, atuadores, nós inteligentes e aplicações incorporadas por meio de uma rede para a interação com estados internos ou ambientes externos. Essa arquitetura utiliza protocolos da Internet para a criação de um novo ecossistema de aplicativos, levando a conexão generalizada de pessoas, serviços, sensores e objetos (Conti *et al.*, 2018; Oppitz; Tomsu, 2018). Atualmente existem diversos domínios equipados com objetos que possuem tecnologia primitiva, sem recursos de comunicação. Ao prover recursos de comunicação para esse objetos é possível melhorar uma gama de aplicações existentes e abre espaço para a criação de novas aplicações IoT, para diferentes tipos de cenários (Shahid; Aneja, 2017). Como por exemplo:

- **Aplicações médicas:** Utiliza-se de dispositivos para o monitoramento contínuo de pacientes, coleta de dados como a pressão arterial, temperatura, entre outras informações possibilitando haver histórico médico mais detalhado e diagnósticos mais assertivos.
- **Aplicações em ambientes inteligentes:** O uso de controle automático de temperatura e iluminação, a possibilidade de controlar fechaduras remotamente e a ajuda de assistentes virtuais em pequenas tarefas podem tornar mais confortável a vida das pessoas.
- **Aplicações agrícolas:** O monitoramento do clima, umidade do ar e do solo, em estufas e campos de cultivo, tem como objetivo auxiliar agricultores a coletar dados significativos para a criação de estratégias e planejamento de novas formas de cultivo mais produtivas e sustentáveis.

Em sistemas IoT objetos comuns transformam-se em objetos inteligentes, capazes de

capturar e compartilhar informações para realizar pequenas tarefas, constituindo aplicações para domínios específicos. Como possuem restrições de processamento e memória podem utilizar recursos de tecnologias independentes de domínios, como Computação em Nuvem (*Cloud Computing*), para tratamento, análise dos dados gerados, entre outros (Al-Fuqaha *et al.*, 2015).

2.2 COMPUTAÇÃO EM NUVEM

A Computação em Nuvem (*Cloud Computing*) é um modelo que fornece recursos de computação de forma conveniente e sob demanda. Esses recursos são altamente configuráveis, providos de maneira fácil e rápida. Dentre esses recursos estão capacidade de armazenamento, processamento e disponibilização de softwares (Foster *et al.*, 2008).

O *National Institute of Standards and Technology* (NIST) definiu três tipos de serviço fornecidos pela Nuvem (Mell; Grance *et al.*, 2011), que são:

- **Software como Serviço (SaaS):** A capacidade de fornecer uma infraestrutura em Nuvem para executar aplicações do cliente. Essas aplicações podem possuir diversos usuários, como *webmails*.
- **Plataforma como serviço (PaaS):** A capacidade de fornecer um ambiente de desenvolvimento de software para o cliente.
- **Infraestrutura como serviço (IaaS):** A capacidade de fornecer recursos de hardware virtualizados, como processamento, armazenamento, sistemas operacionais, entre outros, para o cliente.

A Computação em Nuvem possui diversos desafios, como atraso na comunicação, congestionamento de tráfego, processamento de grande quantidade de dados e custo na comunicação (Mukherjee; Shu; Wang, 2018).

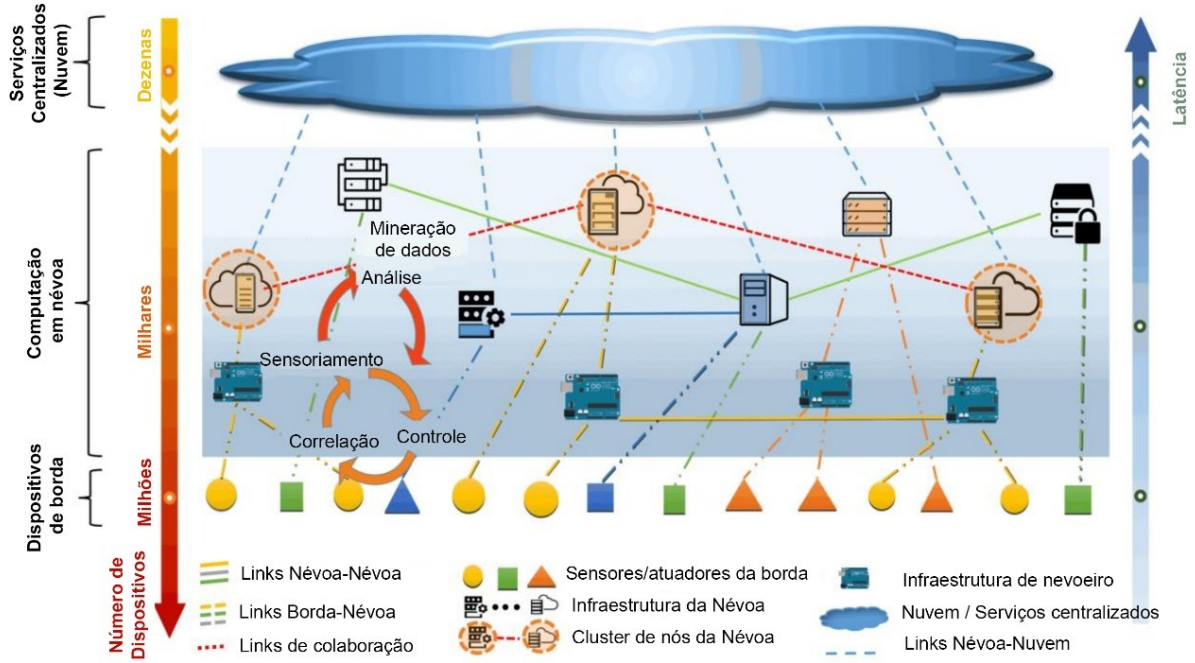
2.3 COMPUTAÇÃO EM NEVOEIRO

A Computação em Nevoeiro (*Fog Computing*) fica localizada próximo a borda da rede, resolvendo diversos problemas de aplicações IoT ligadas diretamente em serviços em Nuvem, possibilitando um novo ecossistema de aplicações (Bonomi *et al.*, 2012). Este é

um conceito, introduzido pela Cisco em 2014, que distribui as funções de armazenamento, processamento, controle e rede para mais próximo do cliente (Chen; Azhari; Leu, 2018).

A Figura 1 mostra como a Computação em Nevoeiro pode dar suporte a um ecossistema amplo baseado em Nuvem para atender dispositivos inteligentes. Com a utilização do Nevoeiro o uso da Nuvem se torna opcional. Diferentes casos de uso podem ter arquiteturas diferentes para dar suporte às funcionalidades dos dispositivos finais.

Figura 1 – Suporte da Computação em Nevoeiro em aplicações IoT e Nuvem.



Fonte: Adaptado de Iorga *et al.* (2018).

Segundo o NIST a Computação em Nevoeiro é um modelo de camadas que permite o acesso a recursos computacionais escaláveis e compartilhados. Este modelo é constituído por nodos *fog*, físicos ou virtuais, que reside entre dispositivos finais e serviços centralizados. A Computação em Nevoeiro facilita a construção de aplicações distribuídas, pois tem consciência da latência, são sensíveis ao contexto, de aplicações e serviços e suportam um sistema comum de gerenciamento de dados (Iorga *et al.*, 2018).

2.4 SISTEMA DE DETECÇÃO DE INTRUSÃO

O primeiro passo para proteger um sistema em rede, é detectar o ataque. Mesmo que não se consiga impedi-lo, poderá fornecer informações valiosas. Por isso a detecção de intrusão é tão importante, e pode ser considerada a primeira linha de defesa em

qualquer sistema de segurança (Kabiri; Ghorbani, 2005). Sistemas de Detecção de Intrusão (*Intrusion Detection System - IDS*) são ferramentas de segurança (Garcia-Teodoro *et al.*, 2009) que tem como objetivo defender um sistema, executando contramedidas ou gerando alertas para uma entidade capaz de realizar ações apropriadas, quando o sistema foi comprometido (Axelsson, 2000).

2.4.1 Método de detecção

Dependendo do tipo de análise realizada, o IDS pode ser classificado em detecção por anomalia ou por assinatura. Na detecção por anomalia o IDS define todo o comportamento padrão e sinaliza todo o comportamento anormal a este padrão definido. Na detecção por assinatura o IDS compara as ações monitoradas com assinaturas definidas anteriormente no sistema. No primeiro, não é necessário ter conhecimento a priori das assinaturas dos ataques como no segundo, mas é preciso estimar o comportamento normal das entidades do sistema. Os dois tipos são semelhantes em termos de operação, mas a detecção baseada em assinatura tende a ser eficaz apenas em ataques conhecidos, em contrapartida detecção baseada em anomalia tem o potencial de detectar eventos intrusivos não visto anteriormente (Garcia-Teodoro *et al.*, 2009; Axelsson, 2000).

2.4.2 Origem da informação analisada

Um IDS também pode ser classificado pela origem da informação analisada. Alguns IDS capturam e analisam pacotes da rede para encontrar ataques, chamados de IDS baseados em redes ou *Network-based Intrusion Detection System* (NIDS), outros IDS analisam informações geradas pelo sistema operacional ou aplicações, chamados de IDS baseado em host ou *Host-based Intrusion Detection System* (HIDS) (Bace; Mell, 2001).

2.4.3 Estratégia de controle

A estratégia de controle define como os elementos, entradas e saídas, de um IDS são controlados e gerenciados. Na estratégia centralizada todo o monitoramento, detecção e geração de relatórios é realizada por um nó central. Na estratégia parcialmente distribuída, os componentes do IDS são distribuídos, mas a detecção da intrusão é realizada por um único nó central. Na estratégia distribuída, é usada uma abordagem orientada a agentes, onde cada nó monitora e realiza a detecção, podendo se comunicar com os outros (Bace; Mell, 2001).

2.4.4 Frequência de execução

A frequência de execução refere-se ao tempo decorrido entre os eventos monitorados e a análise desses eventos. Na execução periódica o IDS é programado para analisar os eventos gerados em intervalos de tempos definidos, sendo o modo utilizado em muitos IDS mais antigo, baseados em *host*, que esperavam registros do sistema operacional. Na execução contínua o IDS monitora o fluxo contínuo de informações recebidos pelo sistema, utilizado em muitos IDS baseados em rede, onde passa um fluxo contínuo de pacotes de rede (Bace; Mell, 2001; Jackson *et al.*, 1999).

2.4.5 Comportamento pós-detecção

Após a identificação da intrusão o IDS pode se comportar de forma ativa ou passiva. No comportamento passivo o IDS é programado para informar o evento detectado para o usuário responsável, através de mensagens consoles, *e-mail*, relatórios, entre outros. No comportamento ativo o IDS é capaz de ser programado pelo usuário a realizar ações pró-ativas e corretivas a um evento crítico identificado, como corrigir a vulnerabilidade do sistema, reconfiguração do *firewall*, entre outros (Jackson *et al.*, 1999).

2.4.6 Sistemas existentes de código aberto

Atualmente existem muitos IDS de código aberto. Um deles é o *Suricata* que é desenvolvido e mantido pela *Open Information Security Foundation*, com foco em segurança, usabilidade e eficiência. Ele possui mecanismo para detecção de intrusão (IDS), prevenção de intrusão (IPS), monitoramento de segurança de rede e processamento de arquivos de pacotes de redes. Existe também o *Snort* que é um software de prevenção de intrusão. Ele conta com um modo *sniffer* que mostra o fluxo de dados recebidos na rede para o usuário, um *log* de pacotes que registra um pacote em disco e um sistema de detecção de intrusão baseado em rede.

2.5 APRENDIZADO DE MAQUINA

A Inteligência Artificial é um campo próspero com muitas aplicações práticas. Principalmente em tarefas que são resolvidas intuitivamente pelas pessoas, mas difíceis de

serem descritas formalmente por um conjunto de regras matemáticas. Dentre elas estão o reconhecimento de palavras faladas e rostos em imagens (Goodfellow *et al.*, 2016).

Aprendizado de máquina é um subconjunto da Inteligência Artificial que constrói um modelo matemático com base em dados e amostras a fim de fazer previsões ou decisões sem serem explicitamente programadas para realizar tal tarefa (Zhang, 2020).

O uso de aprendizagem de máquina para auxiliar na segurança e detecção de intrusão em sistemas IoT tornou-se extremamente importante nos últimos anos. Mesmo assim sistemas tradicionais de aprendizagem de máquina enfrentam dificuldades para detectar pequenas mutações de ataques ao longo do tempo (Costa *et al.*, 2019).

2.5.1 Redes Neurais Artificiais

Segundo (Anderson; McNeill, 1992) as Redes Neurais Artificiais são modelos eletrônicos baseados na estrutura neural do cérebro. Esses modelos foram inspirados na biologia para resolver problemas com os quais a computação tradicional tem dificuldade em lidar. Eles também trazem um conjunto de palavras incomuns para a computação tradicional como comportar, aprender, generalizar, auto-organizar e esquecer.

Segundo (Chua; Yang, 1988) as redes neurais são propostas para resolver problemas em diversas áreas, como otimização, programação linear e não linear, reconhecimento de padrões e visão computacional.

Atualmente aplicações que utilizam redes neurais estão cada vez mais populares. Uma grande vantagem de usar esses modelos é a capacidade de lidar com sistemas naturais complexos que possuem grandes quantidades de informações (Abiodun *et al.*, 2018).

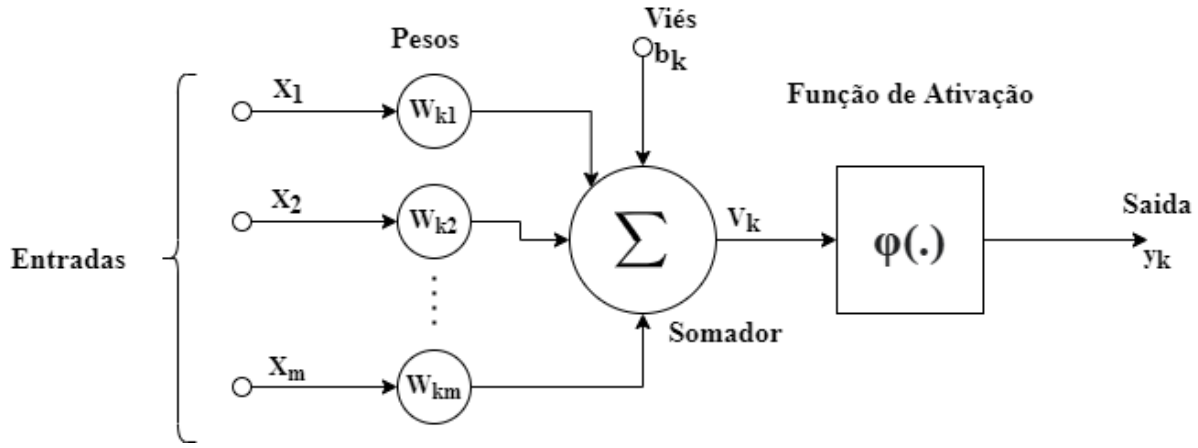
2.5.1.1 Neurônio Matemático

Um neurônio é a unidade de processamento de informação fundamental para o funcionamento de uma rede neural. O neurônio calcula a soma ponderada dos sinais de entradas com seus respectivos pesos. O valor desta soma é aplicado à função de ativação e o seu resultado é transmitido para todas as suas saídas (Haykin *et al.*, 2009). A Figura 2 mostra o modelo deste neurônio.

O neurônio é composto pelos seguintes elementos (Academy, 2019):

- **Sinais de entradas (x_1, x_2, ..., x_m):** São valores externos, que servem de entrada

Figura 2 – Representação de um neurônio matemático.



Fonte: Adaptado de Haykin *et al.* (2009).

para rede;

- **Pesos sinápticos (w_{k1}, w_{k2}, ..., w_{km}):** São valores utilizados para ponderar cada entrada da rede, alterando a intensidade dos valores recebidos. Esses valores serão alterados durante o processo de aprendizado da rede.
- **Combinador Linear ($\sum$):** Agrega as entradas ponderadas por seus respectivos pesos, produzindo um potencial de ativação.
- **Limiar de ativação ou viés (b_k):** Especifica um limiar para o valor produzido pelo combinador linear possa gerar um sinal de ativação.
- **Potencial de ativação (v_k):** Valor produzido pela diferença entre o combinador linear e o limiar de ativação. Se o valor for positivo então o neurônio produziu um potencial excitatório, caso contrário, o potencial será inibitório.
- **Função de ativação (ϕ):** Tem como objetivo limitar o potencial de ativação e decidir se a saída será ativada ou não.
- **Sinal de saída (y_k):** O valor produzido pelo neurônio, que poderá ser conectado às entradas dos neurônios da camada seguinte ou ser o valor de saída da rede.

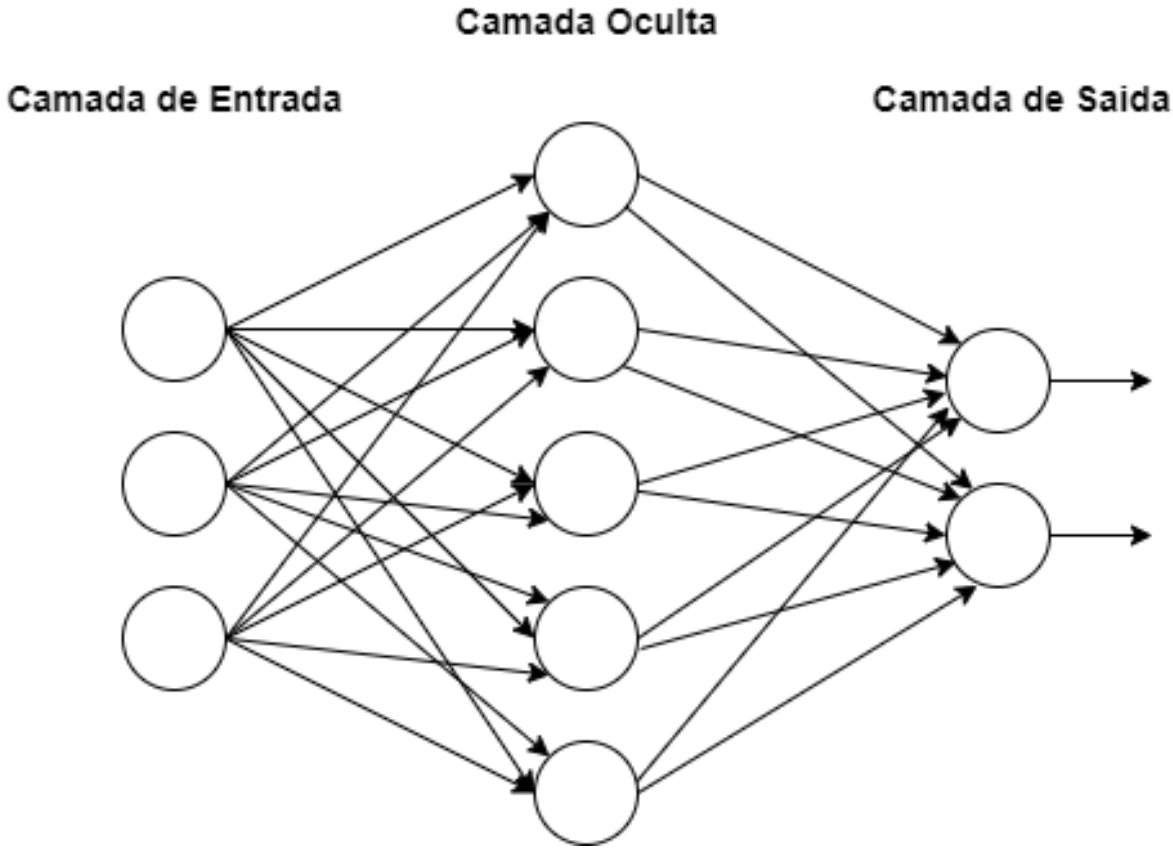
2.5.1.2 Redes Neurais *Feed-Forward*

Redes neurais da classe *Feed-Forward*, são organizadas em camadas, onde os neurônios da camada anterior são conectados aos neurônios da camada seguinte, mas não

vice-versa. As redes neurais *Feed-Forward* não possuem ligações entre neurônios da mesma camada e não possuem ciclos em seus grafos de conexão. Redes que possuem ciclos em seu grafo de conexão são chamadas de Redes Neurais Recorrentes, onde a saída de um neurônio pode servir de entrada para o mesmo, ou para neurônios de camadas anteriores. As redes neurais *Feed-Forward* possuem uma camada de entrada e uma camada de saída. Elas podem possuir camadas ocultas, localizadas entre a camada de entrada e saída. Se a rede possuir mais de uma camada oculta ela é chamada de redes neurais profundas (*deep learning*) (Haykin *et al.*, 2009).

A rede perceptron multicamada (*Multilayer Perceptron* - MLP) é um exemplo de redes neurais *Feed-Forward*, um exemplo dela é mostrado na Figura 3, onde possui três sinais de entrada, uma camada oculta com cinco neurônios e dois neurônios na camada de saída. Essa rede também é chamada de rede totalmente conectada, pois todos os neurônios da camada anterior estão conectados a todos os neurônios da camada seguinte.

Figura 3 – Rede neural perceptron multicamada.



Fonte: O autor.

2.5.1.3 Treinamento supervisionado

No treinamento supervisionado a saída desejada é especificada para cada vetor de entrada da rede. A diferença entre a saída da rede e a saída desejada é tratada como erro. O algoritmo de treinamento é responsável por diminuir o erro entre a saída da rede e a saída desejada (Reed; MarksII, 1999).

A popularidade do treinamento supervisionado para redes perceptrons multicamadas foi aprimorado pelo desenvolvimento do algoritmo *backpropagation* (Haykin *et al.*, 2009). O algoritmo calcula rapidamente as derivadas, que são utilizadas para atualizar os pesos da rede. Este algoritmo pode ser dividido em duas etapas, sendo elas (Academy, 2019):

- ***Forward pass*:** é responsável por propagar a entrada pela rede neural até obter suas previsões (saída da rede)
- ***Backward pass*:** é responsável por calcular o gradiente da função de perda na camada final, sendo este utilizado para atualizar os pesos da rede neural, de forma recursiva. Esta fase é também conhecida como retro-propagação.

2.5.2 Estratégia de decomposição de problemas multiclasse

A classificação multiclasse mapeia um espaço de recurso de entrada com K classes de saídas, $K > 2$. No entanto, muitos classificadores funcionam melhor para problemas de duas classes de saída ($K = 2$). Assim uma abordagem é decompor o problema em K problemas de duas classes. Várias abordagens foram propostas, as mais populares são a *One vs All* (OVA) e *One Vs One* (OVO) (Oong; Isa, 2012)

Na abordagem *One vs All* um problema de K classes é decomposto em K problemas onde K classificadores são criados para discriminar sua respectiva classe em relação a todas as outras.

Na abordagem *One vs One* um problema de K classes é decomposto em pares formando $K(K-1)/2$ problemas, onde cada classificador é responsável por distinguir a classe i da classe j.

3 TRABALHOS CORRELATOS

Neste capítulo serão apresentadas soluções propostas por outros autores na utilização de redes neurais para detecção de intrusão. Os artigos escolhidos apresentam aspectos relacionados com este trabalho em relação a detecção de intrusão utilizando técnicas de redes neurais. Foram usados 5 artigos da base de dados *Institute of Electrical and Electronics Engineers* (IEEE) e 1 artigo da base de dados *Multidisciplinary Digital Publishing Institute* (MDPI).

3.1 THREAT ANALYSIS OF IOT NETWORKS USING ARTIFICIAL NEURAL NETWORK INTRUSION DETECTION SYSTEM

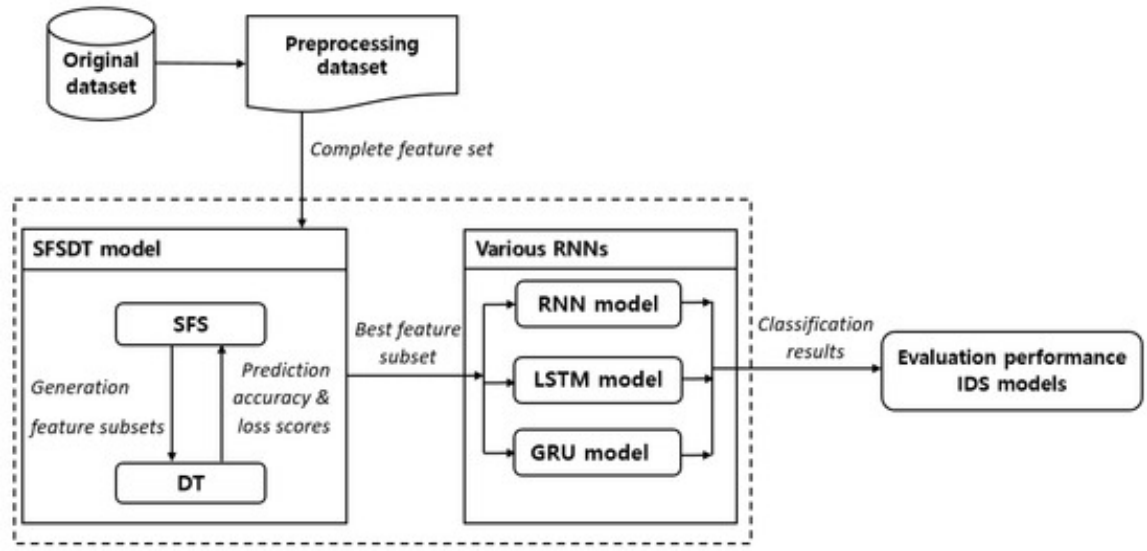
Os autores Hodo *et al.* (2016) propuseram uma rede neural para detecção de intrusão de um sistema IoT. A rede IoT é composta por 1 nodo servidor e 5 nodos sensores. O tráfego é capturado por meio de um *tap* de rede, evitando a modificação dos pacotes. Os autores realizaram ataques de negação de serviço DoS e DDoS, no nodo servidor onde cada ataque teve aproximadamente 10 milhões de pacotes UDP enviados. Eles usaram uma rede neural MLP com 3 camadas, contendo respectivamente 6 neurônios de entrada, 3 na camada oculta e 1 neurônio de saída. A função de ativação *sigmóide* unipolar foi usada nos neurônios da camada intermediária e de saída. Para treinar a rede neural foram utilizadas 2.313 amostras, 496 amostras foram utilizadas para teste e 496 amostras para validação da rede. As amostras são divididas em 2 classes onde 64,18% das amostras são da classe ataque Dos/DDos e 35,82% das amostras são da classe normal. Os autores afirmaram que a rede neural criada obteve 99,4% de precisão. Os autores não especificaram quais informações estavam sendo utilizadas para fazer a análise do tráfego da rede, dificultando a reprodução dos testes.

3.2 NETWORK INTRUSION DETECTION BASED ON NOVEL FEATURE SELECTION MODEL AND VARIOUS RECURRENT NEURAL NETWORKS

Os autores Le *et al.* (2019) observam a alta taxa de falsos positivos nos IDS baseados em anomalias e a baixa precisão nas técnicas avançadas de IDS em ataques do tipo *Remote-to-Local* (R2L) e *User-to-Root* (U2R). Com isso os autores propuseram uma nova estrutura de IDS para melhorar a solução desses problemas.

A estrutura pode ser dividida em 3 partes como mostra a Figura 4. A primeira é um modelo *Sequence Forward Selection Decision Tree* (SFSDT), sendo um algoritmo híbrido *Sequence Forward Selection* (SFS) e um modelo de Árvore de Decisão (*Decision Tree* - DT), para escolher os melhores atributos do conjunto de dados para treinamento. A segunda parte é treinar vários modelos de redes neurais.

Figura 4 – Proposta dos autores Le *et al.* (2019)



Fonte: (Le *et al.*, 2019).

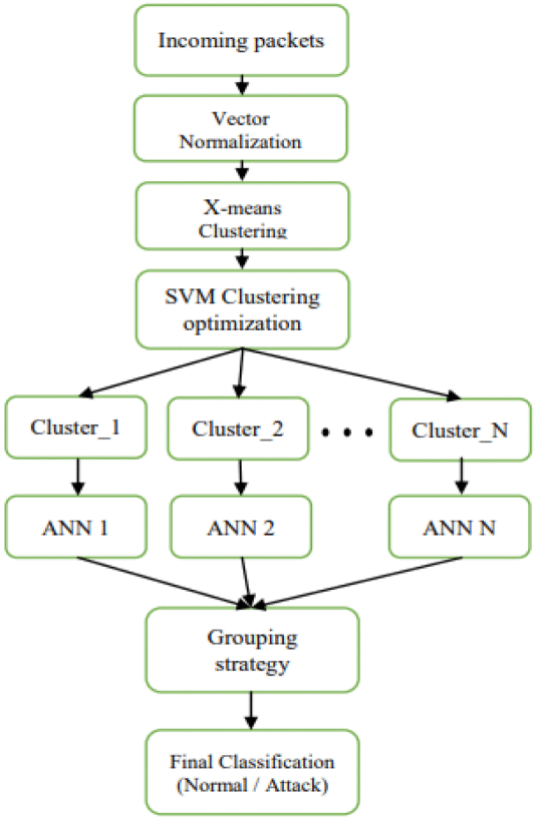
Os autores usaram três modelos, uma Rede Neural Recorrente (RNN), uma rede *Long-Short-Memory* (LSM) e uma rede *Gated Recurrent Unit* (GRU). A terceira parte é a classificação dos modelos treinados. Os autores consideraram a matriz de confusão, a Curva Característica de Operação do Receptor (*Operating Characteristic Curve* - ROC) ou simplesmente curva ROC e o consumo de memória no processo de treinamento para realizar avaliação do modelo. Os autores utilizaram os conjuntos de dados NSL-KDD de 2010 e o ISCX de 2012, especificando os atributos utilizados para realizar os experimentos. Os autores também compararam a estrutura proposta com as de outros autores e o modelo criado por eles apresentou uma precisão superior nos dois conjuntos de dados.

Em particular os modelos GRU e LMS apresentaram uma melhora significativa para a detecção de ataques do tipo R2L e U2R.

3.3 ARTIFICIAL NEURAL NETWORKS OPTIMIZED WITH UNSUPERVISED CLUSTERING FOR IDS CLASSIFICATION

Os autores Lafram, Berbiche & Alami (2019) propuseram a detecção de intrusão por meio de um modelo de rede neural otimizado. Esse modelo é uma combinação de técnicas de aprendizado supervisionada e não supervisionado. Um cenário do conjunto de dados CIC-IDS2017 (Sharafaldin; Lashkari; Ghorbani, 2018) foi utilizado para o treinamento do modelo. A Figura 5 mostra a estrutura proposta.

Figura 5 – Proposta dos autores Lafram, Berbiche & Alami (2019).



Fonte: (Lafram; Berbiche; Alami, 2019)

Primeiramente é realizado o agrupamento do tráfego, onde o modelo inspeciona todo o conjunto de entradas, agrupando os pacotes com comportamentos semelhantes pelo algoritmo *x-means*. Em seguida é usado o algoritmo de Máquina de Vetores de Suporte (*Support Vector Machine* - SVM) para escolher um subconjunto de atributos de cada

grupo. Por fim, é criada uma rede neural MLP para cada grupo e treinada utilizando seus respectivos pacotes e atributos. Os autores apresentam resultados positivos do modelo criado com uma precisão próxima de 100% para as redes neurais de cada grupo e uma diminuição no tempo de execução em comparação com modelos de outros autores citados.

3.4 ARTIFICIAL INTELLIGENCE BASED NETWORK INTRUSION DETECTION WITH HYPER-PARAMETER OPTIMIZATION TUNNING ON THE REALIISTIC CYBER DATASET CSE-CIC-IDS2018 USING CLOUD COMPUTING

Os autores Kanimozhi & Jacob (2019) propuseram utilizar redes neurais para detecção de intrusão de ataques *Botnet*. Os autores utilizaram o conjunto de dados CSE-CIC-IDS2018 (Sharafaldin; Lashkari; Ghorbani, 2018). Foi utilizado uma rede MLP para construir modelo proposto. Uma otimização de hiperparâmetros na rede neural foi usada através da técnica *GridSearchCV*. Os autores fizeram os experimentos utilizando a linguagem *python* e as bibliotecas *scikit-learn* [1] e *tensorflow* [2]. A rede neural foi treinada com 1048575 registros que possuíam 80 atributos utilizando a técnica de amostragem *Cross Validation* com 10 *Fold*. O modelo criado obteve a área abaixo da curva ROC igual 1 e uma precisão próxima a 0.99.

3.5 A NOVEL SNN-ANN BASED IDS IN CLOUD ENVIRONMENT

Os autores Majumder *et al.* (2020) propuseram um modelo híbrido de Redes Neurais Artificial (ANN) e Redes Neurais *Spiking (SNN)* para detecção de intrusão. Os autores usaram o conjunto de dados NSL-KDD. Primeiro é realizada uma classificação binária, onde é usado o modelo SNN para discriminar se o pacote é malicioso. Se o pacote for malicioso, ele é enviado para uma rede neural MLP que classifica o tipo de ataque que ele pertence. Os autores testaram 3 cenários. O primeiro cenário um modelo ANN que obteve 77,83% de precisão. O segundo cenário o modelo SNN que obteve 79,31% de precisão. O terceiro cenário o modelo híbrido ANN-SNN proposto que obteve 85,73% de precisão.

[1] https://scikit-learn.org/
[2] https://www.tensorflow.org/

3.6 The Effective Methods for Intrusion Detection With Limited Network Attack Data: Multi-Task Learning and Oversampling

Os autores Sun *et al.* (2020) propuseram o modelo *Multi-Task Learning* (MTL) para resolver o problema de detecção de intrusão multiclasse. Foram utilizados dois conjuntos de dados, UNSW-NB15 e CSE-CIC-IDS-2018. Foi realizado um pré-processamento de dados nos dois conjunto, removendo registros com valores ausentes e infinitos. Também foi feita a normalização os dados com a função *StanderScaler*, da biblioteca *scikit-Learn*. Foi utilizado o método *SMOTH* para sobreamostrar as amostras minoritárias de modo que a proporção do tráfego normal para os ataques no conjunto de dados de treinamento seja de 1:1. Os autores utilizaram as métricas de precisão, *recall* e *f1-score* para avaliar os modelos. Para comparação, foram realizados experimentos com os modelos SVM, DT, KNN, AdaBoost e BSPL-GN. Resultados experimentais mostram que o método propostos pelos autores têm o melhor desempenho.

3.7 DISCUSSÕES SOBRE O TRABALHO CORRELATO

Ao longo desta seção foram descritos 6 trabalhos correlatos. São diferentes propostas que usaram métodos de aprendizado de máquina para buscar melhorar a detecção de instrução em rede. Destes trabalhos, 3 deles propuseram métodos de detecção binária e 3 deles a detecção multiclasse. A detecção multiclasse é importante, pois fornecem informações mais precisas para o responsável da rede tomar contramedidas eficazes. Em contrapartida é mais difícil obter uma alta taxa de precisão na discriminação das diferentes classes. A área de aprendizado de máquina está se desenvolvendo muito nos últimos anos e tem muitos métodos que podem ser explorados para detecção multiclasse.

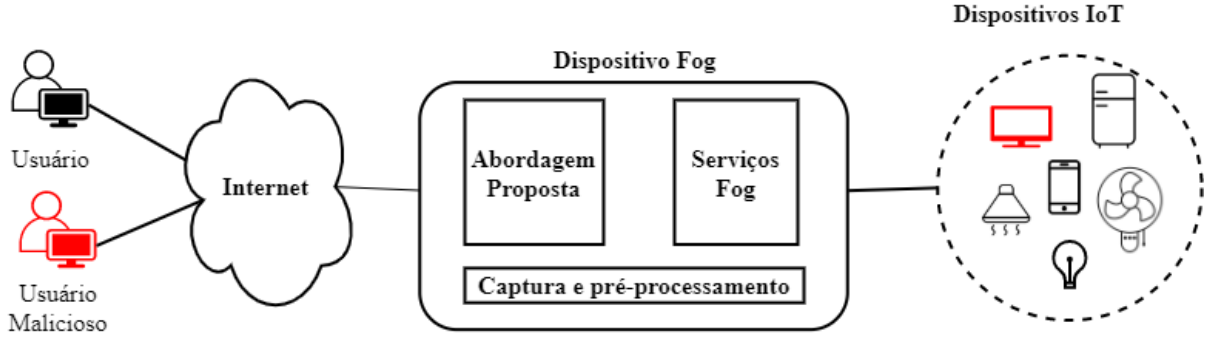
4 ABORDAGEM PROPOSTA

Considerando o crescente aumento de sistemas IoT, suas deficiências em relação a segurança e as pesquisas relacionadas sobre a utilização de redes neurais para detecção de intrusão, este trabalho propõe investigar a utilização da estratégia de decomposição de problemas para melhorar o desempenho de redes neurais na classificação multiclasse no problema de detecção de intrusão.

4.1 CONTEXTUALIZAÇÃO DA PROPOSTA

O trabalho considera o contexto de um sistema IoT que utiliza o dispositivo *fog* para prover serviços e dados para os usuários. Neste contexto, usuários maliciosos podem tentar invadir ou corromper o sistema para roubar dados dos usuários. A Figura 6 mostra a arquitetura desse sistema, onde a abordagem proposta é inserida no dispositivo *fog* para detectar a intrusão. Este trabalho considera que o dispositivo possui mecanismos para captura e pré-processamento do tráfego TCP/IP originados da rede, e além disso, prover os dados necessários para abordagem proposta analisar se está ocorrendo um ataque na rede e que tipo de ataque é esse.

Figura 6 – Visão geral da abordagem proposta.



Fonte: O autor.

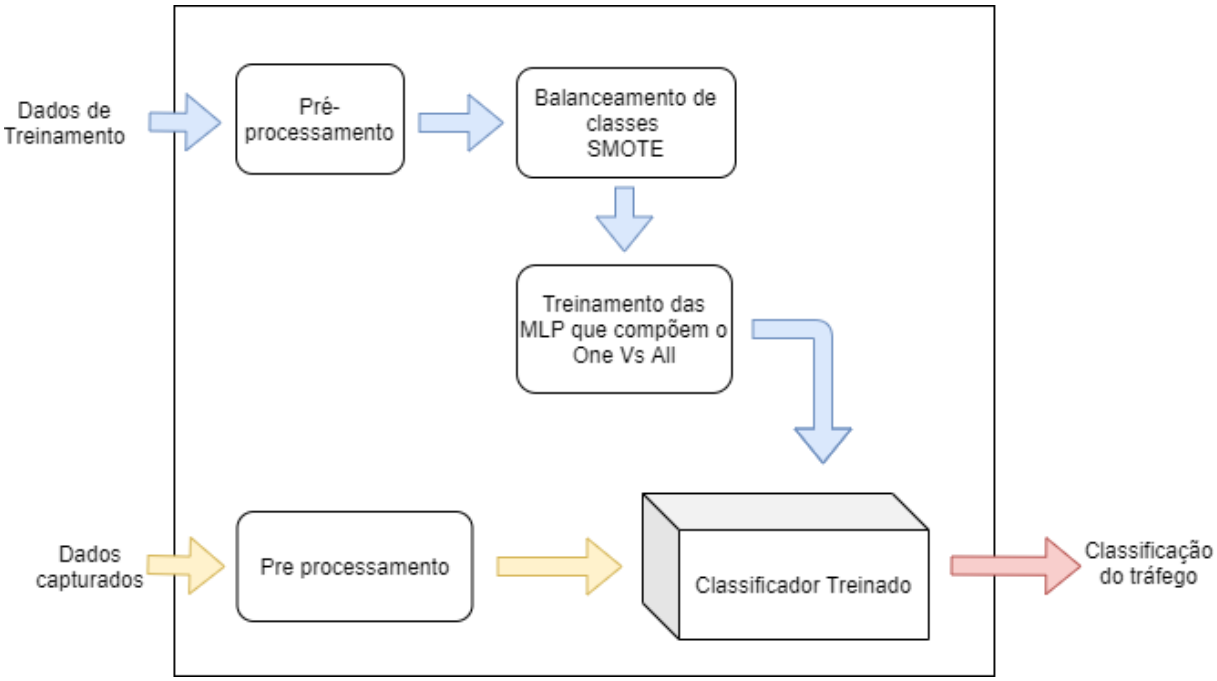
A seguir, são apresentados os principais aspectos da abordagem proposta.

4.2 DESCRIÇÃO DA ABORDAGEM PROPOSTA

Nesta seção são fornecidos maiores detalhes sobre a abordagem proposta para detecção e identificação de intrusões. A Figura 7 ilustra o método proposto neste trabalho. A abordagem necessita de um conjunto de dados para a realização do treinamento do modelos de classificação. Os dados selecionados para o treinamento são formatados e pré-processados para retirada de registros inválidos. Devido a disparidade de proporção de tráfego entre os tipos de ataques, a abordagem conta com uma etapa de balanceamento para que os modelos tenham dados suficientes para compreender os padrões dos ataques que possuem pouca ocorrência. Por fim, é realizado o treinamento da abordagem de classificação gerando um modelo final capaz de classificar novos dados.

Após o processo de treinamento o modelo estará pronto para operar recebendo novos dados capturados da rede para serem analisados e classificados. Durante a execução real os dados são capturados, pré-processados e submetidos para o classificador treinado. O classificador então realiza a detecção e a identificação da categoria do tráfego.

Figura 7 – Método Proposto.

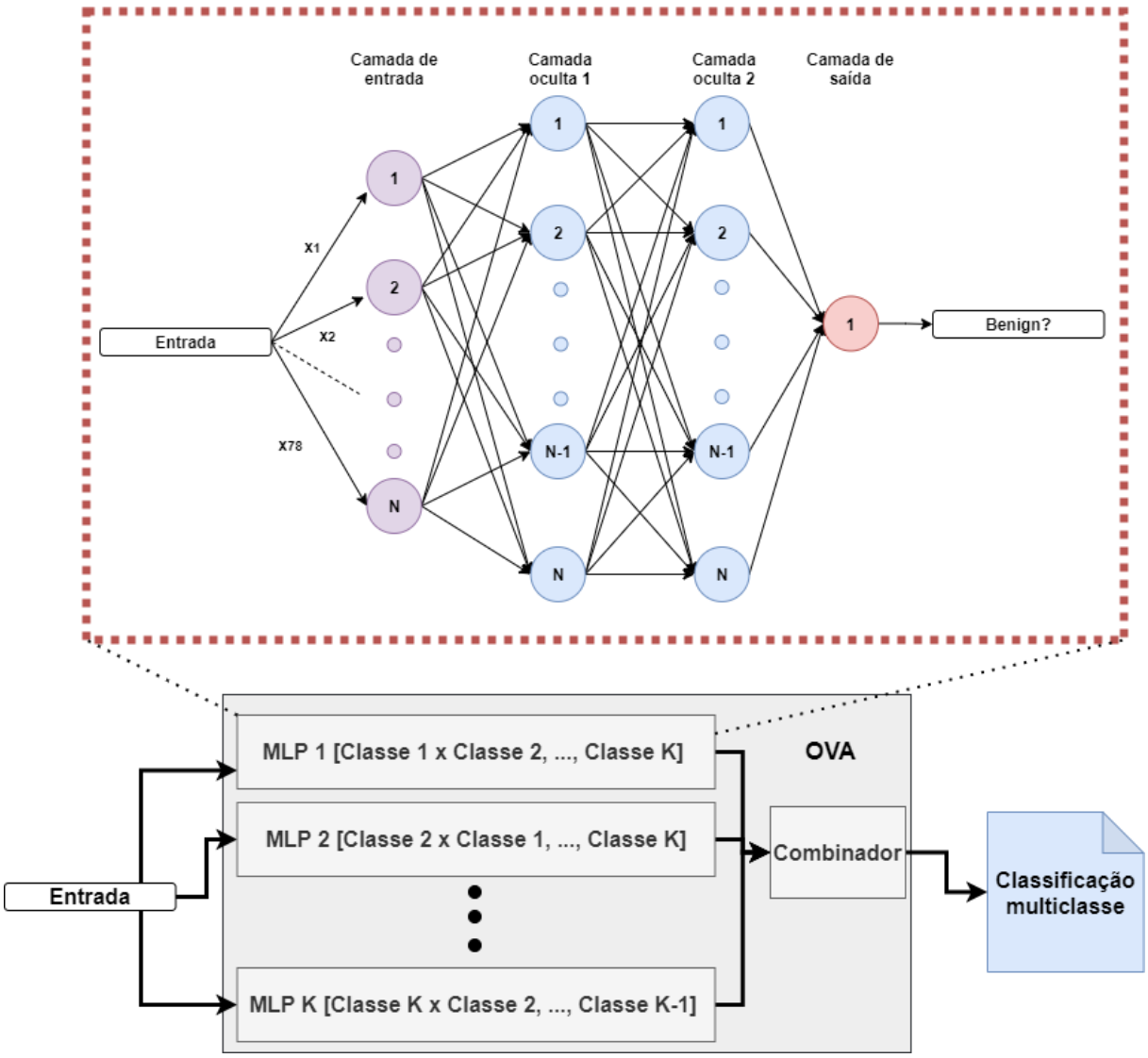


Fonte: O autor.

4.2.1 Classificador

Neste trabalho é investigada a estratégia de decomposição de classes *One Vs All*. Nessa estratégia K classificadores são criados para K classes, onde cada classificador é treinado para discriminar sua respectiva classe (Oong; Isa, 2012). A saída de cada classe é passada para uma função de combinação multiclasse que retorna a saída da rede. A Figura 8 ilustra a abordagem baseada em OVA composta pelos modelos neurais.

Figura 8 – Modelo Proposto.



Fonte: O autor.

A abordagem proposta utiliza modelos neurais perceptron de multicamadas. Conforme pode ser observado na Figura 8, a camada de entrada, assim como, as suas duas camadas ocultas, possuem n neurônios, onde n corresponde a quantidade de atributos do

tráfego considerado. Os neurônios das camadas ocultas, responsáveis pelo aprendizado da rede, utilizam a função de ativação $ReLU$ (Agarap, 2019). Esta função de ativação retorna o valor recebido pelo somador se o mesmo for maior que zero, se os valores forem negativos retorna zero. Por ser uma função simples, tornou-se uma função muito utilizada em diversos tipos de redes neurais, pois é mais fácil de treinar e normalmente atingem bom resultados (Agarap, 2019). A camada de saída é composta por apenas 1 neurônio que utiliza a função de ativação *sigmoide*.

Conforme mencionado anteriormente, o método proposto utiliza K modelos neurais, onde cada um vai discriminar se a entrada pertence ou não a sua respectiva classe. Cada modelo neural é treinado para discriminar sua respectiva classe em relação a todas as outras. A entrada é encaminhada para cada MLP e a saída de cada rede entra em um combinador que classifica a entrada com sendo da classe que teve a rede neural com o maior valor de saída.

Além disso, a abordagem realiza um processo de balanceamento de dados antes do treinamento dos modelos neurais. Esta etapa é realizada com o objetivo de contornar à dificuldade de se trabalhar com dados desbalanceados, onde algumas classes possuem uma quantidade muito superior de registros em relação as demais. Isto faz com que a rede neural tenha um desempenho melhor em classificar as classes com mais registros e classes com poucos registros acabam tendo baixo desempenho da rede. Por isso foi utilizada a técnica de *Synthetic Minority Oversampling Technique* (*SMOTE*) (Bowyer *et al.*, 2011), que consiste em sintetizar os registros falsos para classes que contém poucos registros. Neste trabalho optou-se por sintetizar novos registros em todas as classes de instrução até que elas alcancem a marca de 100.000 registros.

5 AVALIAÇÃO

Nesta seção são descritos alguns aspectos da metodologia utilizada para avaliar a abordagem proposta. Em seguida são apresentados e discutidos os resultados alcançados através dos experimentos.

5.1 EXPERIMENTOS

A metodologia utilizada para realização dos experimentos é descrita a seguir. Inicialmente são apresentadas as características da base de dados e a sua divisão em conjuntos para treino e teste. Em seguida são apresentadas as métricas de classificação consideradas na avaliação. Por fim, são descritos os experimentos realizados.

5.1.1 Base de dados

Neste trabalho foi utilizada a base de dados CSE-CIC-IDS2018[1] (Sharafaldin; Lashkari; Ghorbani, 2018) para realizar os experimentos. A base foi concebida através de um projeto colaborativo entre o *Communications Security Establishment* (CSE) e o *Canadian Institute for Cybersecurity* (CIC). O projeto tem como objetivo desenvolver uma abordagem sistemática para gerar conjunto de dados de instrução em rede diversificado, baseado em perfis de usuários que contém um conjunto de eventos combinado com comportamentos visto na rede.

O conjunto de dados contém os seguintes ataques:

- **Ataque de força bruta:** Ataque que tenta adivinhar a senha de um usuário em sites, usando dicionários específicos com senhas mais usadas e ferramentas para automatizar o processo.

[1] https://registry.opendata.aws/cse-cic-ids2018/

- ***Botnet*:** O atacante usa *malware* do tipo cavalo de troia para tentar controlar as máquinas das vítima.
- **Negação de serviço:** O atacante tenta ocupar todos os recursos do servidor enviado várias requisições para o servidor responder, deixando o serviço indisponível para os usuários.
- **Negação de serviço distribuída:** O atacante usa diversas máquinas para realizar o ataque de negação de serviço.
- **Ataque na web:** São um conjunto de diferentes ataques visando explorar vulnerabilidade de sites *web*, incluindo injeção SQL, injeção de comando e *upload* irrestrito de arquivo.
- **Infiltração na rede:** Neste cenário foi enviado arquivo malicioso para a vítima, que ao ser executado realizou uma varredura na rede interna visando buscar outras vulnerabilidades e explorá-las.

Na Tabela 1 são listados os atributos presentes no conjunto de dados. Esses atributos serão utilizados para treinar a rede neural. Os fluxos TCP são normalmente encerrados após o término da conexão (pelo pacote FIN), enquanto o fluxo UDP é encerrado por um tempo limite de fluxo. A base possui fluxos bidirecionais, onde o primeiro pacote determina as direções *forward* (origem para destino) e para *backward* (destino para origem), daí os 80 recursos estatísticos como duração, número de pacotes, número de bytes, comprimento de pacotes, etc. também são calculados separadamente na direção *forward* e *backward*.

Tabela 1 – Descrição dos atributos existentes na base CSE-CIC-IDS2018.

Atributo	**Descrição**
Flow ID	Identificador do fluxo
Src IP	Endereço IP de origem
Src Port	Porta de origem
Dst IP	Porta de destino
fl_dur	Duração do fluxo
tot_fw_pk	Total de pacotes forward
tot_bw_pk	Total de pacotes backward
tot_l_fw_pkt	Tamanho total dos pacotes forward
fw_pkt_l_max	Tamanho máximo do pacote forward
fw_pkt_l_min	Tamanho mínimo do pacote forward

fw_pkt_l_avg	Tamanho médio do pacote forward
fw_pkt_l_std	Tamanho do desvio padrão do pacote forward
Bw_pkt_l_max	Tamanho máximo do pacote backward
Bw_pkt_l_min	Tamanho mínimo do pacote backward
Bw_pkt_l_avg	Tamanho médio do pacote backward
Bw_pkt_l_std	Tamanho do desvio padrão do pacote backward
fl_byt_s	Número de bytes por segundo no fluxo
fl_pkt_s	Número de pacotes por segundo no fluxo
fl_iat_avg	Tempo médio entre dois fluxos
fl_iat_std	Desvio padrão de tempo de dois fluxos
fl_iat_max	Tempo máximo entre dois fluxos
fl_iat_min	Tempo mínimo entre dois fluxos
fw_iat_tot	Tempo total entre dois pacotes enviados no sendido forward
fw_iat_avg	Tempo médio entre dois pacotes enviados forward
fw_iat_std	Tempo de desvio padrão entre dois pacotes enviados no sentido forward
fw_iat_max	Tempo máximo entre dois pacotes enviados no sentido forward
fw_iat_min	Tempo mínimo entre dois pacotes enviados no sentido forward
bw_iat_tot	Tempo total entre dois pacotes enviados no sentido backward
bw_iat_avg	Tempo médio entre dois pacotes enviados no sentido backward
bw_iat_std	Tempo de desvio padrão entre dois pacotes enviados no sentido backward
bw_iat_max	Tempo máximo entre dois pacotes enviados no sentido backward
bw_iat_min	Tempo mínimo entre dois pacotes enviados no sentido backward
fw_psh_flag	Contador da flag PSH em pacotes no sentido forward
bw_psh_flag	Contador da flag PSH em pacotes no sentido backward
fw_urg_flag	Contador da flag URG em pacotes no sentido forward
bw_urg_flag	Contador da flag URG em pacotes no sentido backward
fw_hdr_len	Total de bytes usados para o header no sentido forward
bw_hdr_len	Total de bytes usados para o header no sentido backward

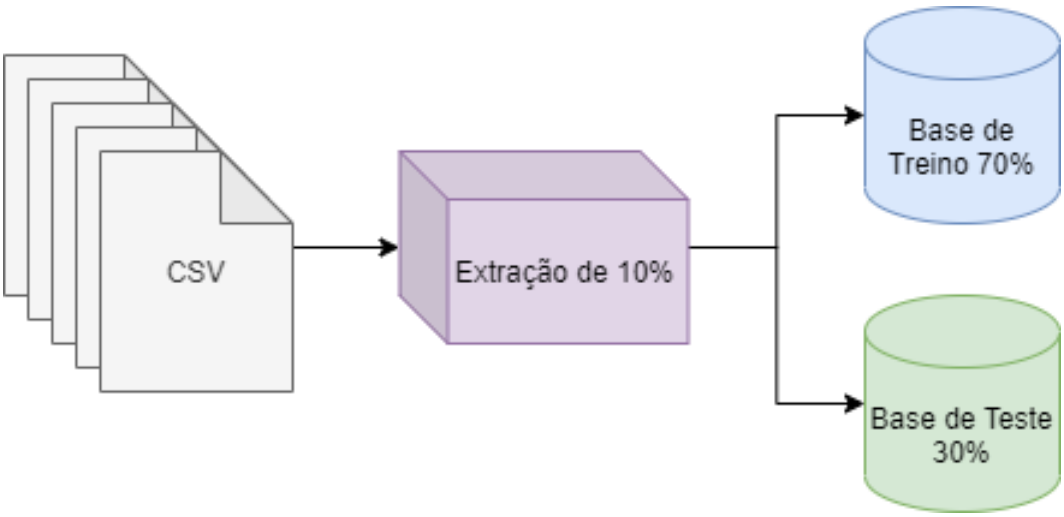
fw_pkt_s	Número de pacotes por segundo no sentido forward
bw_pkt_s	Número de pacotes por segundo no sentido backward
pkt_len_min	Tamanho do menor pacote
pkt_len_max	Tamanho do maior pacote
pkt_len_avg	Média dos tamanhos dos pacotes no fluxo
pkt_len_std	Desvio padrão dos tamanhos dos pacotes no fluxo
pkt_len_va	Variância dos tamanhos dos pacotes no fluxo
fin_cnt	Contador da flag FIN nos pacotes do fluxo
syn_cnt	Contador da flag SYN nos pacotes do fluxo
rst_cnt	Contador da flag RST nos pacotes do fluxo
pst_cnt	Contador da flag PSH nos pacotes do fluxo
ack_cnt	Contador da flag ACK nos pacotes do fluxo
urg_cnt	Contador da flag URG nos pacotes do fluxo
cwe_cnt	Contador da flag CWR nos pacotes do fluxo
ece_cnt	Contador da flag ECE nos pacotes do fluxo
down_up_ratio	Proporção entre download a upload
pkt_size_avg	Média do tamanho dos pacotes
fw_seg_avg	Média de tamanho de segmentos no sentido forward
bw_seg_avg	Média de tamanho de segmentos no sentido backward
fw_byt_blk_avg	Média de bytes de transmissão em massa no sentido forward
fw_pkt_blk_avg	Média de pacotes de transmissão em massa no sentido forward
fw_blk_rate_avg	Média de taxa de transmissão em massa no sentido forward
bw_byt_blk_avg	Média de bytes de transmissão em massa no sentido backward
bw_pkt_blk_avg	Média de pacotes de transmissão em massa no sentido backward
bw_blk_rate_avg	Média de taxa de transmissão em massa no sentido backward
subfl_fw_pk	Número de pacotes em um subfluxo no sentido forward
subfl_fw_byt	Número de bytes em um subfluxo no sentido forward
subfl_bw_pkt	Número de pacotes em um subfluxo no sentido backward
subfl_bw_byt	Número de bytes em um subfluxo no sentido backward
fw_win_byt	Total de bytes enviados na janela inicial no sentido forward
bw_win_byt	Total de bytes enviados na janela inicial no sentido backward

Fw_act_pkt	Nº de pacotes com pelo menos 1 byte de payload de dados TCP no sentido forward
fw_seg_min	Tamanho mínimo de segmento observado no sentido forward
atv_avg	Tempo médio de atividade dos fluxos
atv_std	Desvio padrão do tempo de atividade dos fluxos
atv_max	Tempo máximo de atividade dos fluxos
atv_min	Tempo mínimo de atividade dos fluxos
idl_avg	Tempo médio de ociosidade dos fluxos
idl_std	Desvio padrão do tempo de ociosidade dos fluxos
idl_max	Tempo máximo de ociosidade dos fluxos
idl_min	Tempo mínimo de ociosidade dos fluxos

5.1.2 Pré-processamento da base de dados

A base CSE-CIC-IDS-2018 possui um grande conjunto de dados divididos em 10 arquivos do tipo csv totalizando aproximadamente 6,5 Gb de dados. Devido a enorme quantidade de dados e a dificuldade de recursos para trabalhar com a base completa ouve a necessidade de utilizar um subconjunto de 10% da base para realizar os experimentos. Sendo que o simples fato de realizar o carregamento da base no plataforma, esgotava a quantidade de memória RAM disponível. Essa extração de 10% dos registros foi realizada de forma aleatória, mas mantendo a proporção da quantidade de registros por classes. A Figura 9 apresenta um diagrama ilustrando o processo.

Figura 9 – Pré-processamento da base de dados



Fonte: O autor.

Inicialmente cada arquivo csv passou por um pré-processamento. Nesta etapa foram removidos alguns atributos que apenas alguns arquivos continham, sendo: Flow ID, Src IP, Src Port, Dst IP. Em seguida também foi realizada a remoção de registros inválidos. Foram removidos os registros com valores nan e inf.

Além disso, foi realizado o agrupamento das classes por tipo de ataque. Originalmente os registros eram classificados pelo tipo e ferramenta de ataque que foi utilizada. A Tabela 2 mostra como foi feito o novo agrupamento. Posteriormente, as novas classes foram transformadas para categorias numéricas, de modo, a tornar possível o treinamento dos modelos neurais.

Tabela 2 – Agrupamento das classes

Classes Agrupadas	**Classes Originais**
Benign	*Benign*
Bot	*Bot*
Brute Force	*SSH-Bruteforce*
	FTP-BruteForce
Web Attack	*Brute Force -Web*
	Brute Force -XSS
	SQL Injection
DDOS	*DDOS attack-HOIC*
	DDOS attack-LOIC-UDP
	DDoS attacks-LOIC-HTTP
DoS	*DoS attacks-GoldenEye*
	DoS attacks-Slowloris
	DoS attacks-Hulk
	DoS attacks-SlowHTTPTest
Infiltration	*Infiltration*

Outro processo realizado no pré-processamento da base foi a normalização dos dados. A padronização do conjunto de dados ajuda a melhorar alguns classificadores baseados em aprendizado de maquina que precisam que seus recursos individuais estejam normalmente distribuído. O método utilizado para a normalização dos dados foi o escalonamento padrão, dado pela Equação 5.1, sendo x a amostra, u a média e s o desvio padrão. A média e o desvio padrão são obtidos com base na estatística de cada atributo (coluna) do conjunto de dados.

$$z = \frac{x - u}{s} \tag{5.1}$$

Em seguida foi utilizado a técnica de *Hold Out*, onde há uma divisão de dados entre treino e teste. Dessa forma, foram usados 70% da base para treinamento e 30% para

o teste. A Tabela 3 mostra o número total de registro de cada classe, a frequência total, o número de registros dos conjuntos de treinamento e de teste.

Tabela 3 – Recursos da base dados

	Classe	**Frequência**	**Nº Total**	**Nº Treino**	**Nº Teste**
0	*Benign*	0.829785	1339059	937341	401718
1	*Bot*	0.017735	28619	20033	8586
2	*Brute Force*	0.023607	38095	26667	11428
3	*DDOS*	0.078324	126394	88476	37918
4	*DoS*	0.040546	65430	45801	19629
5	*Infiltration*	0.009947	16052	11236	4816
6	*Web Attack*	0.000058	93	65	28

5.1.3 Métricas de Avaliação

Para avaliação do modelo serão utilizadas algumas métricas de avaliação comumente empregadas para avaliar aplicações de classificação. Inicialmente, a partir dos experimentos são obtidas as seguintes medidas :

- **Verdadeiro Positivo (VP):** Instâncias classificadas corretamente como intrusão pelo método de detecção.
- **Verdadeiro Negativo (VN):** Instâncias classificadas corretamente como normal pelo método de detecção.
- **Falso Positivo (FP):** Instâncias classificadas como intrusão pelo método de detecção, mas que na realidade são instâncias normais.
- **Falso Negativo (FN):** Instâncias classificadas como normal pelo método de detecção, mas que na realidade são instâncias de intrusão.

A partir das medidas supracitadas é possível calcular seguintes métricas de avaliação (Almiani *et al.*, 2020):

- **Acurácia (ACC):** Proporção de instâncias classificadas corretamente sobre o número total de classificação. Ela é calculada através da Equação 5.2.

$$ACC = \frac{VP + FP}{VP + VN + FP + FN} \tag{5.2}$$

- **Precisão:** Apresenta a taxa de instâncias classificadas corretamente como intrusão sobre o total de instâncias classificadas intrusão. Ela pode ser obtida através da Fórmula 5.3.

$$PRECISÃO = \frac{VP}{VP + FP} \tag{5.3}$$

- **Sensibilidade (*Recall*):** Apresenta a taxa de instâncias classificadas como intrusão sobre o total de instâncias intrusão. Ela pode ser calculada através da Fórmula 5.4.

$$RECALL = \frac{VP}{VP + FN} \tag{5.4}$$

- **Especificidade (*True Negative Rate - TNR*) :** Proporção classificada corretamente instancias normais sobre o total de instâncias normais. Ela pode ser calculada através da Fórmula 5.5.

$$TNR = \frac{FP}{FP + FV} \tag{5.5}$$

- ***F1-score*:** Esta pontuação é para avaliar a qualidade geral do modelo, calculando a média harmônica entre a precisão e o recall. Ela pode ser obtida através da Formula 5.6.

$$f1\_score = 2 * \frac{PRECISAO * RECALL}{PRECISAO + RECALL} \tag{5.6}$$

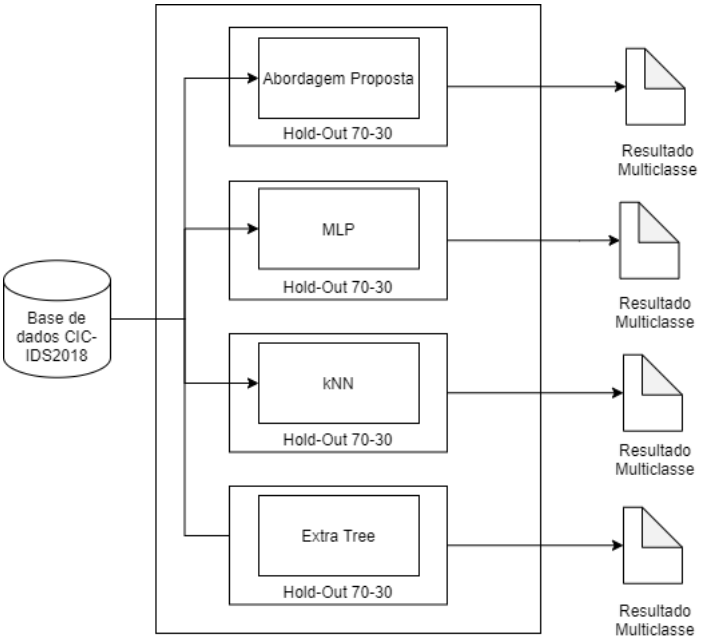
- **Acurácia balanceada:** Métrica utilizada em classificação multiclasse onde os dados são desbalanceados, onde a acurácia da classe é ponderada pelo número de instâncias. Pode ser obtido pela Fórmula 5.7.

$$ACC\_BAL = \frac{RECALL + TNR}{2} \tag{5.7}$$

5.1.4 Experimentos Realizados

Para fins de comparação e avaliação da abordagem proposta, foram realizados experimentos com a solução proposta e com outras 3 abordagens de classificação. Sendo eles um modelo MLP, um modelo *ExtraTree* (ET) e uma *K-Nearest Neighbor* (kNN). A Figura 10 mostra os experimentos realizados com os 4 modelos. Após o pré-processamento descrito na seção 5.1.2, foi realizado o treinamento dos 4 modelos com o conjunto de teste.

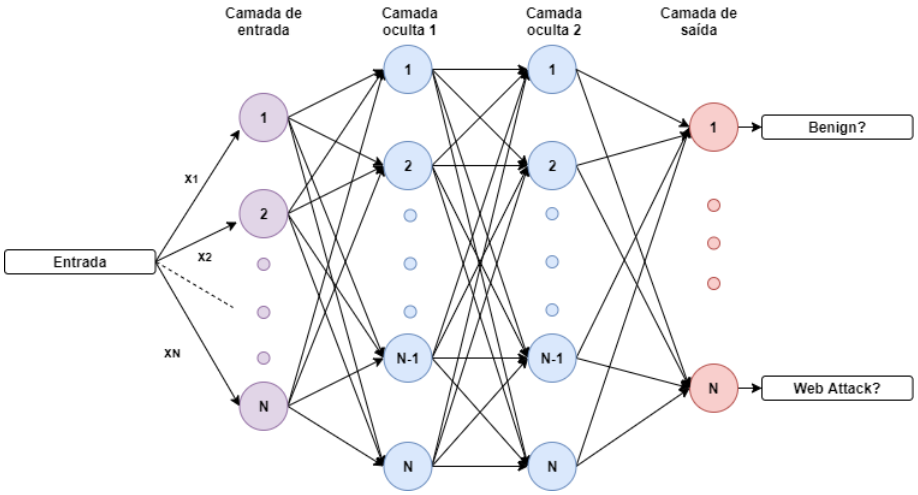
Figura 10 – Experimentos realizados



Fonte: O autor.

A *Multilayer Perceptron* (MLP) utilizada para comparação é similar ao modelo neural que compõe a solução proposta. A camada de entrada, assim como, as suas duas camadas ocultas, possuem 78 neurônios com função de ativação *relu*. No entanto, a camada de saída possui 7 neurônios com função de ativação *softmax*, um para cada classe, onde cada neurônio indica se a entrada pertence ou não a respectiva classe. A entrada vai pertencer a classe do neurônio que possui o maior valor. A Figura 11 ilustra a estrutura do modelo.

Figura 11 – Modelo MLP treinado



Fonte: O autor.

O *K—Nearest Neighbors* (KNN) é um algoritmo muito utilizado na área de aprendizado de máquina de mineração de dados (*data mining*). A classificação é realizada através da comparação de similaridade em relação aos dados de treinamento. A classe majoritária entre os K registros mais similares é atribuída ao dado a ser classificado. Nesse experimento, utilizou-se K com valor 1.

Por fim, a *Extra Trees* (ET) consiste na criação de várias árvores de decisões aleatórias. Em um problema de classificação cada Árvore de Decisão vai classificar a entrada, votando em uma classe. A classe que receber mais votos será a saída do modelo.

5.1.5 Materiais utilizados

Neste trabalho foi utilizado a plataforma do *Google Colaboratory* para realização dos experimentos. É uma plataforma em nuvem que disponibiliza um ambiente de execução em *python*. A máquina possui o processador AMD EPYC 7B12 com 13GB de memória RAM. O sistema operacional é Ubuntu 18.4 LTS.

Os experimentos foram realizados na linguagem *python* versão 3.7. Foram utilizadas as seguintes bibliotecas:

- *Pandas*: é uma biblioteca para manipulação e analise de dados, que foi utilizada para carregar a base de dados e realizar processamento dos dados.
- *Scikit-learn*: é uma biblioteca que possui muitas ferramentas para auxiliar o processo de criação e treinamento de modelos de aprendizado de maquina. Foi utilizada para implementações dos modelos treinandos, funções de separação dos dados entre treino e teste, além da função para aplicação da técnica SMOTE na base de dados.

5.2 RESULTADOS

Os resultados das quatro abordagens foram obtidos utilizando o conjunto de teste que contém registros que não foram usados pelos modelos na fase de treinamento. Cada modelo descrito na Figura 10 foi testado e os resultados obtidos foram utilizados para calcular as métricas descritas na Seção 5.1.3.

A Tabela 4 apresenta o resultado dos experimentos com a abordagem proposta, são apresentados os resultados de cada classe e a avaliação geral do modelo. Os resultados obtidos mostram-se satisfatórios. A abordagem alcançou *recall* de 0,99 para o tráfego

Tabela 4 – Resultados do modelo proposto OVA.

Classe	***acc***	***loss***	***precision***	***recall***	***TNR***	***f1-score***
Benign	0,99	0,01	0,99	0,99	0,95	0,99
Bot	1	0	1	1	1	1
Brute Force	0,99	0,01	0,83	0,95	1	0,89
DDOS	1	0	1	1	1	1
DoS	0,99	0,01	0,97	0,89	1	0,92
Infiltration	0,99	0,01	0,31	0,23	0,99	0,27
Web Attack	1	0	0,08	0,93	1	0,15
Acurácia				**0,98**		
Acurácia Balanceada				**0,85**		
Precisão				**0,98**		

normal. A métrica *recall* indica a porcentagem de tráfego normal que foi identificado como benigno dentre todos os registros de tráfego normal presentes no conjunto de teste. Além disso, apresentou altas taxas de *recall* para a detecção e identificação de ataques *Botnet*, DDoS e *Brute Force*, nos três casos apresentou *recall* superior a 0,95. Para os ataques DoS, a abordagem apresentou uma boa taxa de detecção, aproximadamente 0,89, conforme pode ser observado na Tabela 4. No entanto, a abordagem não foi capaz de alcançar uma boa taxa de detecção para a classe de ataques *Infiltration*, apresentando um *recall* de apenas 0,23. Para a classe *Web Attack* a abordagem foi capaz de alcançar uma taxa de *recall* de 0,93. No entanto, apresentou uma precisão baixa, e este fato refletiu na baixa taxa de *F1-score*, métrica que consiste em uma média harmônica entre precisão e *recall*.

No geral, a abordagem proposta apresentou uma acurácia extremamente alta, o motivo se deve principalmente ao fato de ter apresentado dificuldades na detecção apenas das classes que possuíam poucos registros, as classes *Infiltration* e *Web Attack*. Desse modo, a taxa de acurácia geral não foi prejudicada pelo desempenho ruim em relação a estas classes. A acurácia balanceada, apresentada também na Tabela 4, permite mensurar de modo mais realístico o desempenho. A abordagem proposta alcançou 0,85 de acurácia balanceada.

Na Tabela 5 são apresentados os resultados alcançados pelo modelo MLP único no experimento. Assim como a abordagem proposta, o modelo MLP foi capaz de obter ótimas taxas de *recall* na identificação de tráfego normal e de ataques *Botnet*, *Brute Force*, DDoS e DoS. O modelo teve problemas com as classes *Infiltration* e *Web Attack*, apresentando *recall* de 0,02 e 0,43, e *F1-score* de 0,04 e 0,56, respectivamente. De modo geral, o modelo neural foi capaz de obter uma acurácia balanceada de 0,75.

Tabela 5 – Resultados do modelo MLP.

Classe	acc	loss	precision	recall	TNR	f1-score
Benign	0,99	0,01	0,99	1	0,94	0,99
Bot	1	0	1	1	1	1
Brute Force	0,99	0,01	0,84	0,94	1	0,89
DDOS	1	0	1	1	1	1
DoS	0,99	0,01	0,96	0,89	1	0,93
Infiltration	0,99	0,01	0,5	0,02	1	0,04
Web Attack	1	0	0,8	0,43	1	0,56
Acurácia				**0,98**		
Acurácia Balanceada				**0,75**		
Precisão				**0,98**		

Os resultados obtidos pelo algoritmo kNN são apresentados na Tabela 6. Assim como as abordagens anteriores, o algoritmo apresentou resultados positivos para a maioria das classes. Ele também encontrou dificuldades para identificar ataques *Infiltration* e *Web Attack*, obtendo um *F1-score* de 0,14 e 0,67 respectivamente. No geral, esta abordagem alcançou uma acurácia de 0,78, superando o modelo neural. No entanto, cabe destacar, que este algoritmo realiza muitas comparações em tempo de detecção, o que pode levar a um *delay* maior de detecção.

Tabela 6 – Resultados do modelo kNN.

Classe	*acc*	*loss*	*precision*	*recall*	*TNR*	*f1-score*
Benign	0,98	0,02	0,99	0,99	0,95	0,99
Bot	1	0	1	1	1	1
Brute Force	0,99	0,01	0,99	0,75	1	0,85
DDOS	1	0	1	1	1	1
DoS	0,99	0,01	0,87	1	0,99	0,93
Infiltration	0,98	0,02	0,14	0,13	0,99	0,14
Web Attack	1	0	0,74	0,61	1	0,67
Acurácia				**0,98**		
Acurácia Balanceada				**0,78**		
Precisão				**0,98**		

Por fim, a Tabela 7 apresenta os resultados do experimento efetuado com o modelo *Extra Tree*. Assim como as demais abordagens, ele alcançou ótimas taxas de *recall* para os ataques *Botnet*, *Brute Force*, DDoS e DoS. As principais dificuldades também estão relacionadas com com os ataques *Infiltration* e *Web Attack* com *recall* de 0,10 e 0,50, respectivamente. A acurácia balanceada apresentada pela Extra Tree foi de 0,78.

Tabela 7 – Resultados do modelo ET.

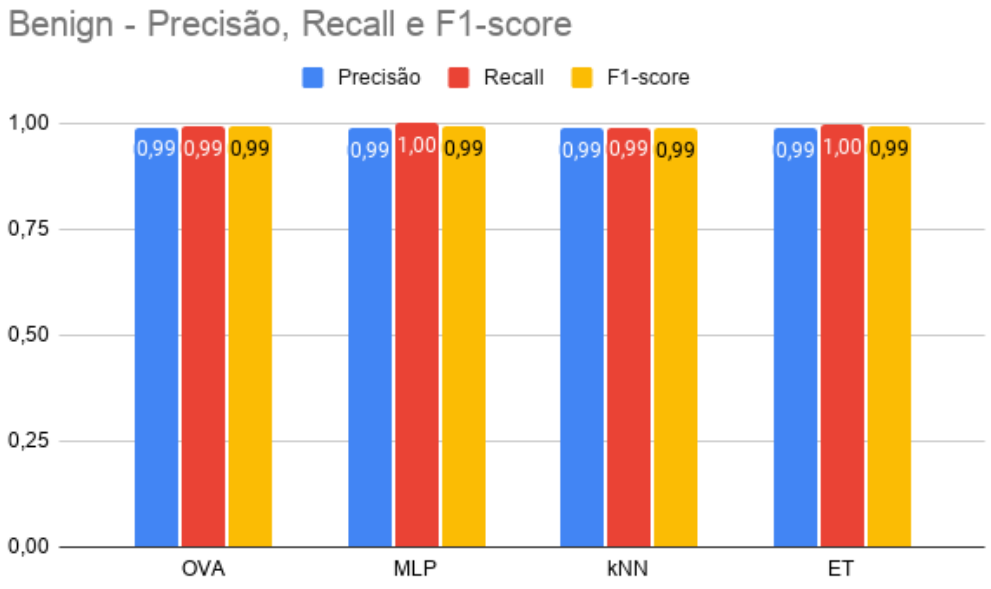
Classe	*acc*	*loss*	*precision*	*recall*	*TNR*	*f1-score*
Benign	0,99	0,01	0,99	1	0,95	0,99
Bot	1	0	1	1	1	1
Brute Force	0,99	0,01	0,84	0,94	1	0,89
DDOS	1	0	1	1	1	1
DoS	0,99	0,01	0,96	0,89	1	0,93
Infiltration	0,99	0,01	0,21	0,1	1	0,14
Web Attack	1	0	0,93	0,5	1	0,65
Acurácia				**0,98**		
Acurácia Balanceada				**0,78**		
Precisão				**0,98**		

5.3 DISCUSSÕES

Nesta seção, serão discutidos os resultados e comparados os desempenhos dos diferentes modelos em relação a cada uma das classes observadas. Para isso foram criados gráficos comparado os dados de precisão, *recall* e *f1-score* de cada uma das classes.

O gráfico apresentado na Figura 12 exibe uma comparação dos modelos na identificação do tráfego normal. Os modelos não tiveram dificuldade na detecção dessa classe, alcançando resultados próximos a 1, nas três métricas avaliadas.

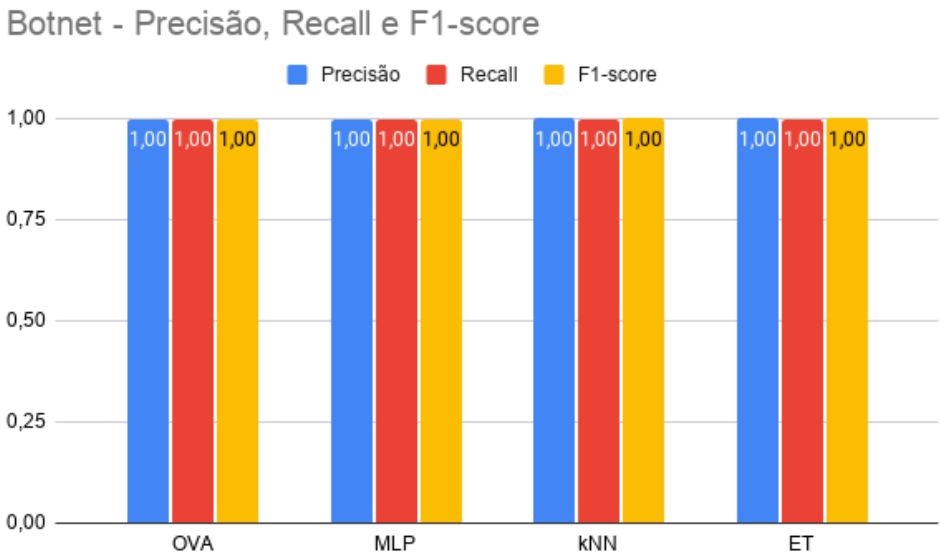
Figura 12 – Gráfico comparativo de detecção do tráfego normal.



Fonte: O autor.

A comparação dos modelos em relação aos ataques *Botnet* é ilustrada na Figura 13. Todos os modelos obtiveram resultados bons, alcançando também resultados próximos a 1, nas três métricas avaliadas.

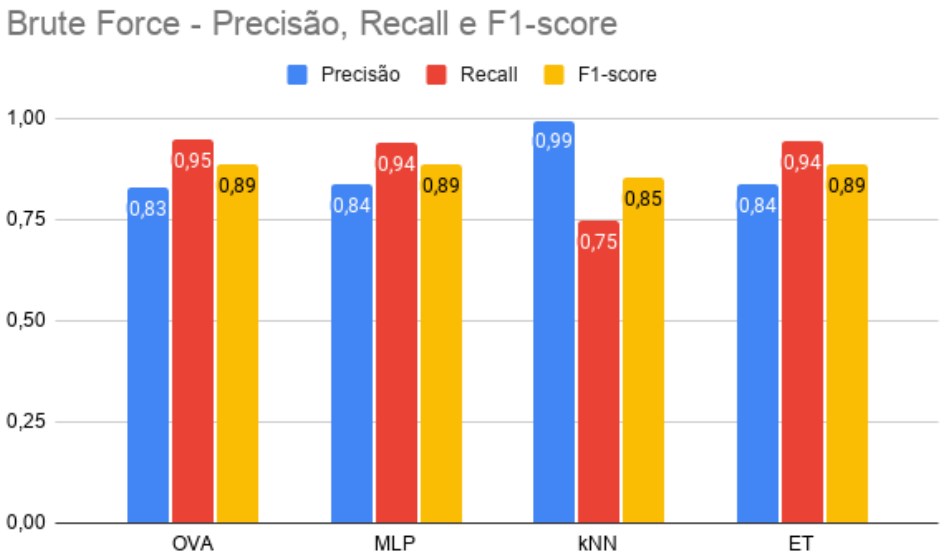
Figura 13 – Gráfico comparativo de detecção de ataques Botnet.



Fonte: O autor.

Na Figura 14 é apresentado o gráfico comparativo em relação as taxas de identificação de ataques do tipo *Brute Force* obtidas pelos 4 modelos.

Figura 14 – Gráfico comparativo de detecção de ataques *Brute Force*.

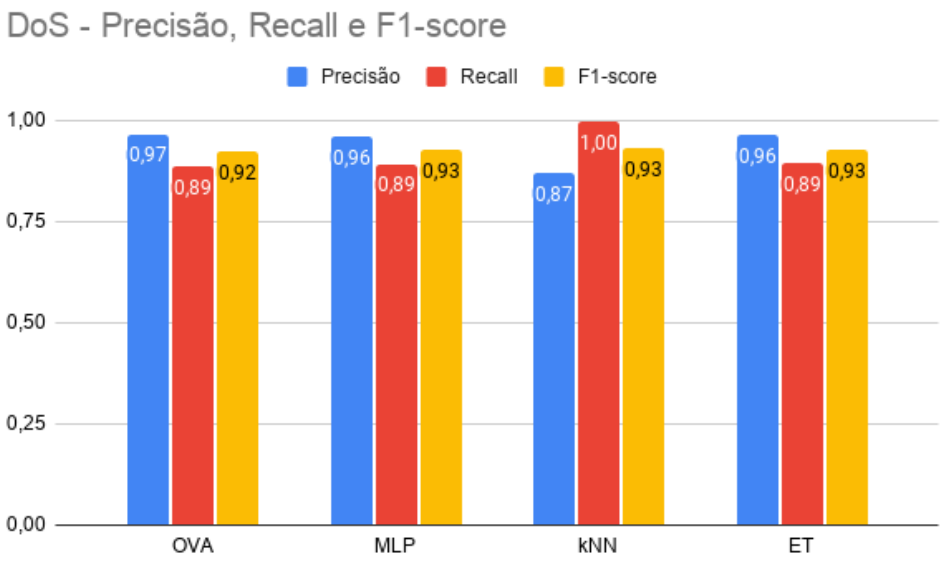


Fonte: O autor.

O KNN obteve a maior precisão (0,99), mas obteve o pior *recall* (0,75), deixando de detectar 25% dos ataques *Brute Force*. Sendo que no geral todos obtiveram o *f1-score* próximo a 0,89. A abordagem proposta OVA foi capaz de obter o maior *recall* (0,95) e apresentou uma precisão, similar ao modelo ET e ao modelo MLP.

Em relação aos ataques DoS, os resultados obtidos pelas abordagens são comparados no gráfico apresentado na Figura 15. A abordagem proposta, obteve a maior precisão (0,97), o modelo kNN alcançou o maior *recall* (1,00) e todos os modelos ficaram com f1-score próximo a 0,93.

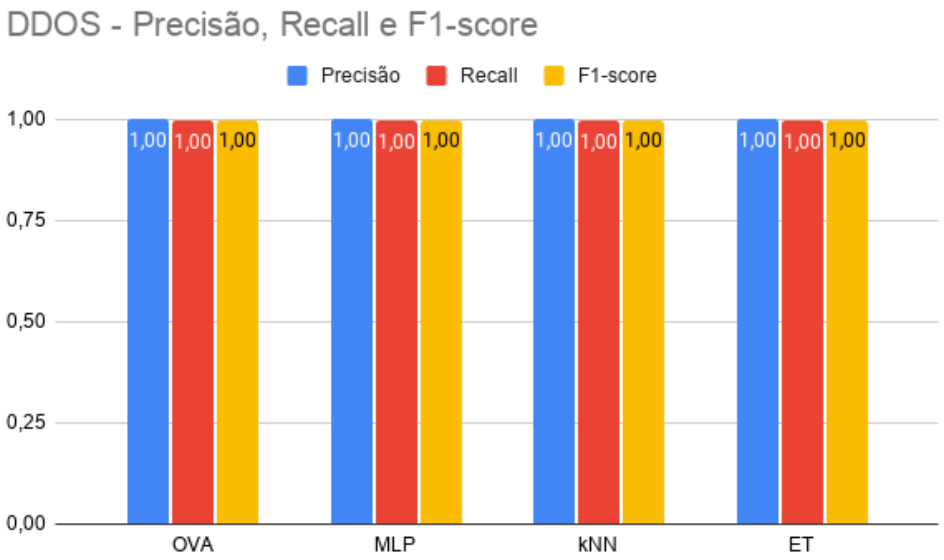
Figura 15 – Gráfico comparativo de detecção de ataques *DoS*.



Fonte: O autor.

Assim como, na detecção de trafego normal e de ataques *Botnet*, todas as abordagens foram capazes de alcançar ótimas taxas de precisão e *recall* na identificação de ataques DDoS, conforme pode ser observado na Figura 16.

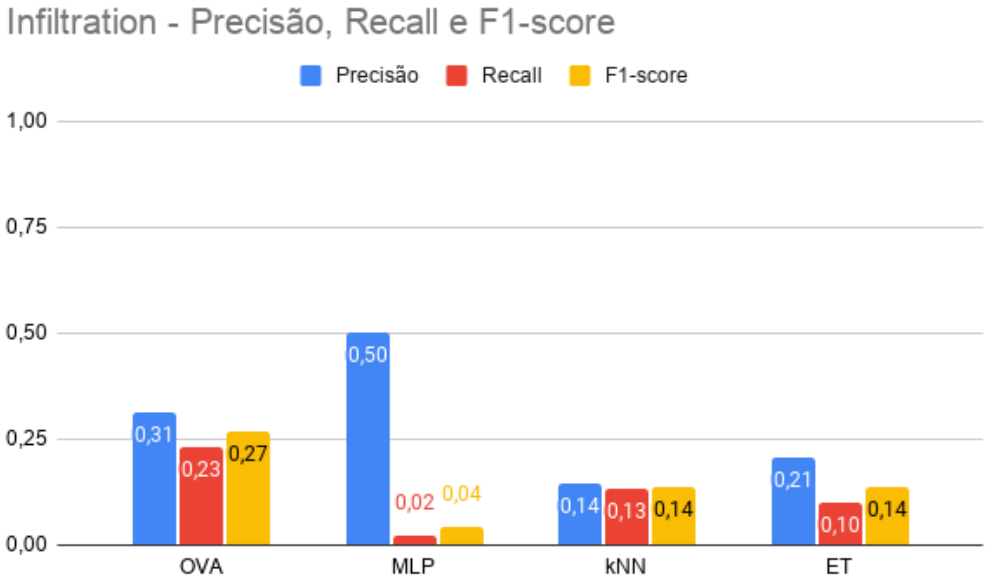
Figura 16 – Gráfico comparativo de detecção de ataques *DDoS*.



Fonte: O autor.

Na Figura 17 é apresentado o gráfico comparativo sobre o desempenho dos modelos em relação a detecção de ataques *Infiltration*.

Figura 17 – Gráfico comparativo de detecção de ataques *Infiltration*.



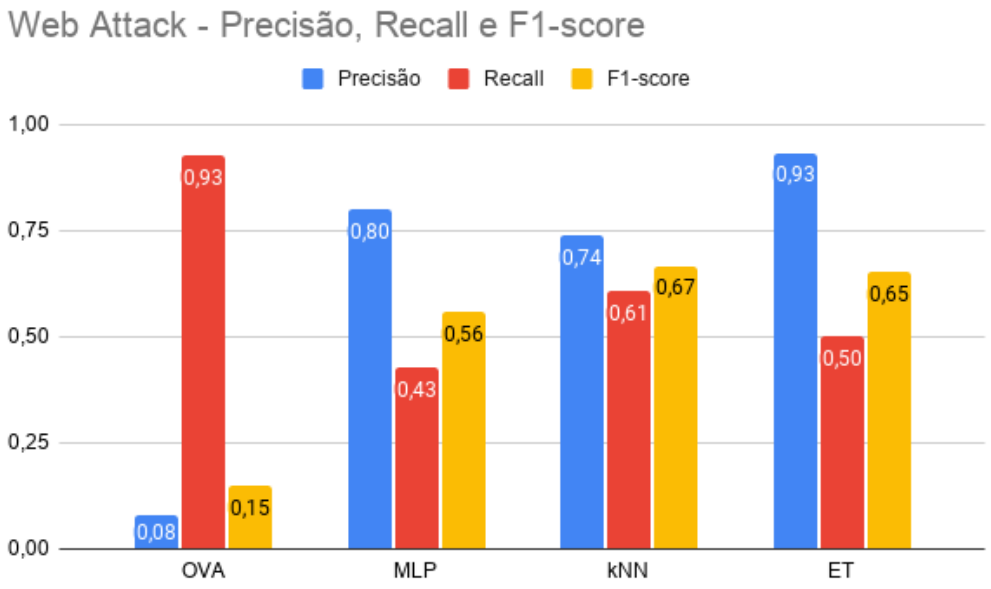
Fonte: O autor.

Todas as abordagens tiveram dificuldade em relação a este tipo de ataque, apresentando desempenho muito ruins. Destaque para o modelo MLP que obteve a melhor

precisão (0,50), o pior *recall* (0,02) e o pior *f1-score* (0,04). A proposta obteve o melhor *recall* e melhor f1-score (0,27), mas mesmo assim é um valor muito abaixo do ideal.

A Figura 18 apresenta o gráfico de comparação entre o desempenho dos modelos em relação a classe *Web Attack*. A abordagem proposta apresentou uma taxa de *recall* muito superior às demais para este tipo de ataques, no entanto, sua precisão foi baixa, indicando que apesar de ter identificado muitos *Web Attack*, ela também apresentou alta taxa de falsos positivos. As demais técnicas, apesar de alcançarem um *recall* menor, apresentaram uma maior harmonia entre precisão e recall. No entanto, as taxas de detecção ainda são consideradas baixas.

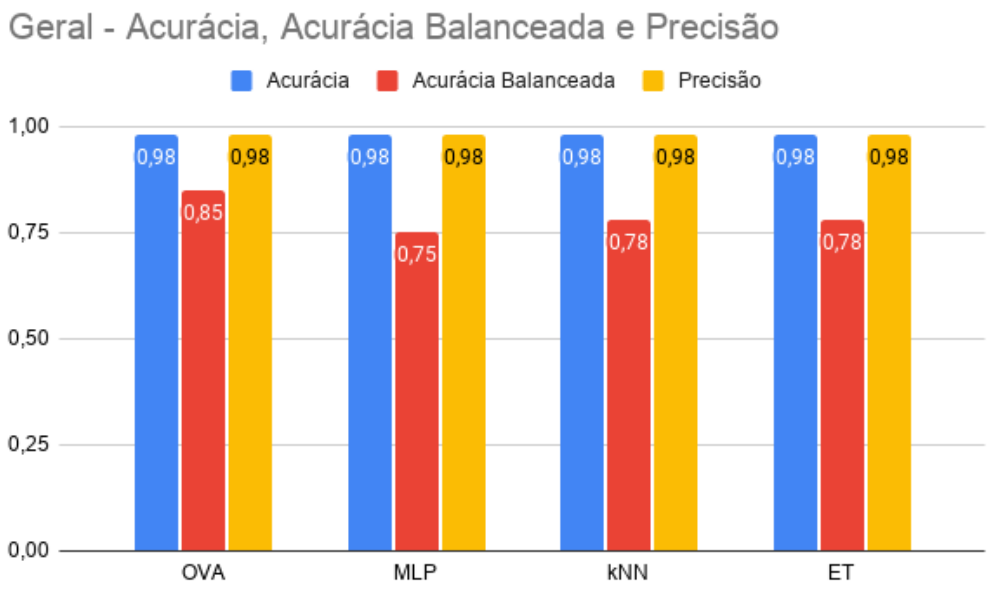
Figura 18 – Gráfico comparativo de detecção de ataques *Web Attack*.



Fonte: O autor.

A acurácia alta presente nos resultados das abordagens se justifica pelo fato delas terem alcançado um ótimo desempenho na detecção do tráfego normal que possui uma grande quantidade de registros. No entanto, ao se analisar a acurácia balanceada, métrica que pondera as classes através da proporção de tráfego, é possível observar que este número tem uma queda. As abordagens MLP, kNN e *Extra Tree* obtiveram 0,75, 0,78 e 0,78 de acurácia balanceada, respectivamente. A abordagem proposta foi capaz de melhorar essa métrica, alcançando uma acurácia balanceada de 0,85. Portanto, nas métricas gerais, a abordagem foi capaz de obter um desempenho interessante, superando as demais.

Figura 19 – Gráfico comparação do desempenho geral dos modelos.



Fonte: O autor.

A Tabela 8 mostra uma comparação dos resultados da abordagem proposta e do trabalho dos autores Sun *et al.* (2020).

Tabela 8 – Comparação com trabalho do estado da arte.

	Modelo Proposto (OVA)			**Sun *et al.* (2020)**		
	PRE	**Recall**	**F1-Score**	**PRE**	**Recall**	**F1-Score**
Classe Benign	**0,99**	**0,99**	**0,99**	-	-	-
Classe Bot	**1,00**	**1,00**	**1,00**	0,53	0,88	66,42
Classe Brute Force	**0,83**	**0,95**	**0,89**	0,58	0,94	71,49
Classe DDOS	**1,00**	**1,00**	**1,00**	0,97	**1,00**	98,17
Classe DoS	**0,97**	**0,89**	**0,92**	0,90	0,88	88,95
Classe Infiltration	0,31	0,23	0,27	**0,54**	**0,74**	**62,13**
Classe Web Attack	0,08	**0,93**	0,15	**0,56**	0,90	**68,96**

É possível observar que a abordagem proposta neste trabalho foi capaz de obter ótimos resultados e superar a outra abordagem em todas as métricas para as classes *Botnet*, *Brute Force*, *DDoS* e *DoS*. Por poutro lado, em relação a classe de ataques *Infiltration*, o desempenho da proposta de Sun *et al.* (2020) alcançou melhores resultados. Em relação a classe *Web Attacck*, a abordagem proposta alcançou o maior *recall*, porém teve uma baixa precisão, indicando taxa de falsos positivos alta. Cabe destacar aqui que o trabalho de Sun *et al.* (2020) não apresentou os resultados para a identificação de tráfego normal.

Métricas baixas nesse quesito podem indicar alta taxa de bloqueio de tráfego normal, que não deveria ser bloqueado, portanto, é uma métrica muito importante também. Não foi possível realizar a comparação deste trabalho com os demais trabalhos do estado da arte pois eles utilizavam classificação binara ou utilizavam outras bases para fazer a pesquisa.

No geral, a abordagem proposta neste trabalho apresentou um bom resultado. Alcançando excelentes taxas de detecção na maioria dos ataques avaliados. No entanto, não foi capaz de apresentar melhoras no desempenho de detecção de ataques como *Infiltration* e *Web Attack*. Portanto, são necessários maiores estudos na aplicação de técnicas de aprendizado de máquina na detecção e classificação de intrusão em redes IoT.

6 CONCLUSÃO

Os sistemas IoT estão cada vez mais comuns e presentes na vida das pessoas, podendo ser aplicados em diversas áreas e diferentes contextos. Entretanto, por causa da baixa capacidade de processamento e memória apresentam uma séries de preocupações de segurança. Este trabalho teve como objetivo propor uma abordagem para detecção e identificação de intrusões baseado no método *One Vs All* e em modelos neurais MLP. Além disso, a abordagem contou com a técnica SMOTE para criar novos registros para balancear a base.

A avaliação da abordagem proposta foi realizada através da técnica *Hold Out* e com o conjunto de dados CSE-CIC-IDS2018, sendo um grande conjunto de dados recentes que vem sendo utilizando em pesquisas nessa área. Foram realizados experimentos para avaliar a abordagem proposta em relação as técnicas de aprendizado de máquina. No geral todos os modelos tiveram um bom desempenho, destacando o modelo proposto que obteve uma acurácia balanceada de 0,85% sendo a maior em comparação aos modelos. No entanto, todas as abordagens tiveram dificuldade em identificar os ataques *Infiltration* e *Web Attack*. Como trabalhos futuros, é importante destacar que são necessários maiores estudos na aplicação de técnicas de aprendizado de maquina na detecção e classificação de intrusão em redes IoT.

REFERÊNCIAS

ABIODUN, Oludare Isaac; JANTAN, Aman; OMOLARA, Abiodun Esther; DADA, Kemi Victoria; MOHAMED, Nachaat AbdElatif; ARSHAD, Humaira. State-of-the-art in artificial neural network applications: A survey. **Heliyon**, Elsevier, v. 4, n. 11, p. e00938, 2018.

ACADEMY, Data Science. **Deep Learning Book**. 2019. Disponível em: <http://www-.deeplearningbook.com.br>.

AGARAP, Abien Fred. **Deep Learning using Rectified Linear Units (ReLU)**. 2019.

AL-FUQAHA, Ala; GUIZANI, Mohsen; MOHAMMADI, Mehdi; ALEDHARI, Mohammed; AYYASH, Moussa. Internet of things: A survey on enabling technologies, protocols, and applications. **IEEE communications surveys & tutorials**, IEEE, v. 17, n. 4, p. 2347–2376, 2015.

ALMIANI, Muder; ABUGHAZLEH, Alia; AL-RAHAYFEH, Amer; RAZAQUE, Abdul. Cascaded hybrid intrusion detection model based on som and rbf neural networks. **Concurrency and Computation: Practice and Experience**, Wiley Online Library, v. 32, n. 21, p. e5233, 2020.

ANDERSON, Dave; MCNEILL, George. Artificial neural networks technology. **Kaman Sciences Corporation**, v. 258, n. 6, p. 1–83, 1992.

AXELSSON, Stefan. **Intrusion detection systems: A survey and taxonomy**. [S.l.], 2000.

BACE, Rebecca; MELL, Peter. **NIST special publication on intrusion detection systems**. [S.l.], 2001.

BONOMI, Flavio; MILITO, Rodolfo; ZHU, Jiang; ADDEPALLI, Sateesh. Fog computing and its role in the internet of things. In: **Proceedings of the first edition of the MCC workshop on Mobile cloud computing**. [S.l.: s.n.], 2012. p. 13–16.

BOWYER, Kevin W.; CHAWLA, Nitesh V.; HALL, Lawrence O.; KEGELMEYER, W. Philip. SMOTE: synthetic minority over-sampling technique. **CoRR**, abs/1106.1813, 2011.

CHEN, Yan-Da; AZHARI, Muhammad Zulfan; LEU, Jenq-Shiou. Design and implementation of a power consumption management system for smart home over fog-cloud computing. In: IEEE. **2018 3rd International Conference on Intelligent Green Building and Smart Grid (IGBSG)**. [S.l.], 2018. p. 1–5.

CHUA, Leon O; YANG, Lin. Cellular neural networks: Theory. **IEEE Transactions on circuits and systems**, IEEE, v. 35, n. 10, p. 1257–1272, 1988.

CONTI, Mauro; DEHGHANTANHA, Ali; FRANKE, Katrin; WATSON, Steve. **Internet of Things security and forensics: Challenges and opportunities**. [S.l.]: Elsevier, 2018.

COSTA, Kelton AP da; PAPA, João P; LISBOA, Celso O; MUNOZ, Roberto; ALBUQUERQUE, Victor Hugo C de. Internet of things: A survey on machine learning-based intrusion detection approaches. **Computer Networks**, Elsevier, v. 151, p. 147–157, 2019.

FOSTER, Ian; ZHAO, Yong; RAICU, Ioan; LU, Shiyong. Cloud computing and grid computing 360-degree compared. In: IEEE. **2008 grid computing environments workshop**. [S.l.], 2008. p. 1–10.

FRUSTACI, Mario; PACE, Pasquale; ALOI, Gianluca; FORTINO, Giancarlo. Evaluating critical security issues of the iot world: Present and future challenges. **IEEE Internet of things journal**, IEEE, v. 5, n. 4, p. 2483–2495, 2017.

GARCIA-TEODORO, Pedro; DIAZ-VERDEJO, Jesus; MACIÁ-FERNÁNDEZ, Gabriel; VÁZQUEZ, Enrique. Anomaly-based network intrusion detection: Techniques, systems and challenges. **computers & security**, Elsevier, v. 28, n. 1-2, p. 18–28, 2009.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron; BENGIO, Yoshua. **Deep learning**. [S.l.]: MIT press Cambridge, 2016.

HAYKIN, Simon *et al.* Neural networks and learning machines. **Upper Saddle River: Pearson Education**, v. 3, 2009.

HODO, Elike; BELLEKENS, Xavier; HAMILTON, Andrew; DUBOUILH, Pierre-Louis; IORKYASE, Ephraim; TACHTATZIS, Christos; ATKINSON, Robert. Threat analysis of iot networks using artificial neural network intrusion detection system. In: IEEE. **2016 International Symposium on Networks, Computers and Communications (ISNCC)**. [S.l.], 2016. p. 1–6.

HUGHES, Kieran; MCLAUGHLIN, Kieran; SEZER, Sakir. Dynamic countermeasure knowledge for intrusion response systems. In: IEEE. **2020 31st Irish Signals and Systems Conference (ISSC)**. [S.l.], 2020. p. 1–6.

IORGA, M; FELDMAN, L; BARTON, R; MARTIN, MJ. **Fog Computing Conceptual Model. Special Publication (NIST SP)-500–325**. [S.l.]: NIST, 2018.

JACKSON, Kathleen A *et al.* Intrusion detection system (ids) product survey. **Los Alamos National Laboratory**, 1999.

KABIRI, Peyman; GHORBANI, Ali A. Research on intrusion detection and response: A survey. **IJ Network Security**, v. 1, n. 2, p. 84–102, 2005.

KANIMOZHI, V; JACOB, T Prem. Artificial intelligence based network intrusion detection with hyper-parameter optimization tuning on the realistic cyber dataset cse-cic-ids2018 using cloud computing. In: IEEE. **2019 International Conference on Communication and Signal Processing (ICCSP)**. [S.l.], 2019. p. 0033–0036.

LAFRAM, Ichrak; BERBICHE, Naoual; ALAMI, Jamila El. Artificial neural networks optimized with unsupervised clustering for ids classification. In: IEEE. **2019 1st International Conference on Smart Systems and Data Science (ICSSD)**. [S.l.], 2019. p. 1–7.

LE, Thi-Thu-Huong; KIM, Yongsu; KIM, Howon *et al.* Network intrusion detection based on novel feature selection model and various recurrent neural networks. **Applied Sciences**, Multidisciplinary Digital Publishing Institute, v. 9, n. 7, p. 1392, 2019.

MAJUMDER, Debojit; SINGH, Anubhav; GHOSH, Partha; PHADIKAR, Santanu. A novel snn-ann based ids in cloud environment. In: IEEE. **2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)**. [S.l.], 2020. p. 913–918.

MELL, Peter; GRANCE, Tim *et al.* The nist definition of cloud computing. Computer Security Division, Information Technology Laboratory, National . . . , 2011.

MUKHERJEE, Mithun; SHU, Lei; WANG, Di. Survey of fog computing: Fundamental, network applications, and research challenges. **IEEE Communications Surveys & Tutorials**, IEEE, v. 20, n. 3, p. 1826–1857, 2018.

OONG, Tatt Hee; ISA, Nor Ashidi Mat. One-against-all ensemble for multiclass pattern classification. **Applied Soft Computing**, Elsevier, v. 12, n. 4, p. 1303–1308, 2012.

OPPITZ, Marcus; TOMSU, Peter. Internet of things. In: **Inventing the Cloud Century**. [S.l.]: Springer, 2018. p. 435–469.

REED, Russell; MARKSII, Robert J. **Neural smithing: supervised learning in feedforward artificial neural networks**. [S.l.]: Mit Press, 1999.

SHAHID, Noman; ANEJA, Sandhya. Internet of things: Vision, application areas and research challenges. In: IEEE. **2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)**. [S.l.], 2017. p. 583–587.

SHARAFALDIN, Iman; LASHKARI, Arash Habibi; GHORBANI, Ali A. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: **ICISSp**. [S.l.: s.n.], 2018. p. 108–116.

SOE, Yan Naung; FENG, Yaokai; SANTOSA, Paulus Insap; HARTANTO, Rudy; SAKURAI, Kouichi. Implementing lightweight iot-ids on raspberry pi using correlation-based feature selection and its performance evaluation. In: SPRINGER. **International Conference on Advanced Information Networking and Applications**. [S.l.], 2019. p. 458–469.

Sun, L.; Zhou, Y.; Wang, Y.; Zhu, C.; Zhang, W. The effective methods for intrusion detection with limited network attack data: Multi-task learning and oversampling. **IEEE Access**, v. 8, p. 185384–185398, 2020.

VIKRAM, N; HARISH, KS; NIHAAL, MS; UMESH, Raksha; KUMAR, Shetty Aashik Ashok. A low cost home automation system using wi-fi based wireless sensor network incorporating internet of things (iot). In: IEEE. **2017 IEEE 7th International Advance Computing Conference (IACC)**. [S.l.], 2017. p. 174–178.

ZHANG, Xian-Da. Machine learning. In: **A Matrix Algebra Approach to Artificial Intelligence**. [S.l.]: Springer, 2020. p. 223–440.

Apêndices

A Artigo

Esta seção contém o artigo resultante deste trabalho no formato da SBC (Sociedade Brasileira de Computação).

Abordagem de Detecção e Identificação de Intrusão baseada em Redes Neurais no contexto de Internet das Coisas

João Vitor Cardoso[1],

[1]Departamento de Informática – Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – CEP 88040-900 – Florianópolis – SC – Brazil

joao.v.c@grad.ufsc.br

Abstract. *Internet of Things (IoT) is a paradigm present in applications in several areas, such as medical, agriculture, home automation, among others. Due to resource restrictions, these devices may have security-related vulnerabilities. Currently, studies using neural networks to solve classification problems are obtaining positive results, including in the context of intrusion detection. In this work the problem decomposition strategy called One vs All will be investigated. This strategy explores the high rate of precision that binary classifiers have and uses them to solve multiclass problems. This work proposes to use this strategy in conjunction with models of feedfoward neural networks for intrusion detection in IoT systems.*

Resumo. *Internet das Coisas (IoT) é um paradigma presente em aplicações de diversas áreas, tais como área médica, agricultura, automação residencial, dentre outras. Devido às restrições de recursos, estes dispositivos podem apresentar vulnerabilidades relacionadas à segurança. Atualmente, trabalhos utilizando redes neurais para solucionar problemas de classificação estão obtendo resultados positivos, inclusive no contexto de detecção de intrusão. Neste trabalho será investigada a estratégia de decomposição de problemas chamada One vs All. Esta estratégia explora a alta taxa de precisão que os classificadores binários possuem e os utiliza para resolver problemas multiclasse. Este trabalho propõe usar esta estratégia em conjunto com modelos de redes neurais feedfoward para detecção de intrusão em sistemas IoT.*

1. Introdução

Atualmente, existem diversos dispositivos eletrônicos e eletrodomésticos que podem ser conectados à Internet, disponibilizando dados e serviços remotos para os usuários [Vikram et al. 2017]. Por exemplo, existem tomadas e lâmpadas que podem ser ligadas à rede sem fio, sendo possível ligá-las e desligá-las remotamente, agendando horário para tais funcionalidades. Esses dispositivos podem ser utilizados para projetar ambientes IoT, como automação residencial.

Ambientes IoT são compostos por dispositivos com recursos restritos, baixa capacidade de processamento e memória limitada. Esses dispositivos normalmente possuem sensores que coletam informações do ambiente, gerando grande quantidade de dados. Para realizar o armazenamento, processamento e análise desses dados são utilizados sistemas em nuvem. O grande tráfego gerado por esses dispositivos e a latência causada pela distância entre o servidor em Nuvem e os dispositivos IoT é um grande problema

para esses sistemas. Por isso surgiu o paradigma de *Fog Computing* que adiciona um nó intermediário fazendo um pré-processamento antes de enviar os dados para a nuvem [Iorga et al. 2018].

Conforme supracitado, os dispositivos IoT possuem baixa capacidade de processamento e isso dificulta a adição de mecanismo de segurança [Frustaci et al. 2017]. Casos de violação de segurança são comuns em sistemas IoT, pois muitas vezes a segurança não recebe a devida atenção no projeto destes sistemas.

A Internet das Coisas está cada vez mais presente no dia a dia das pessoas por meio de aparelhos usados no seu cotidiano, mas ainda possui muitas vulnerabilidades. Em 2016, muitos dispositivos IoT foram invadidos, através do *malware Mirai*, e utilizados para fazer ataques de negação de serviço, prejudicando algumas funcionalidades de serviços, como *Twitter*, *Spotify*, tornando-os indisponíveis. A principal motivação deste trabalho é melhorar a classificação do comportamento intrusivo em sistemas que estão cada vez mais próximo das pessoas.

Uma abordagem para prover segurança a estes sistemas é tentar identificar ataques utilizando mecanismos como Sistema de Detecção de Intrusão baseado em rede. Este tipo de ferramenta analisa o tráfego, sem interferir na comunicação entre os dispositivos, para descobrir se um ataque está sendo realizado [Soe et al. 2019, Garcia-Teodoro et al. 2009].

Além de detectar a ocorrência do ataque é importante também identificar o tipo de ataque que está ocorrendo para auxiliar na tomada de decisão do responsável da rede. Segundo os autores [Hughes et al. 2020] Sistemas de Resposta de Intrusão, que visam responder a ataques automaticamente assim que são alertados, necessitam de conhecimentos específicos para aplicar contramedidas precisas e eficazes.

Este trabalho tem como objetivo geral investigar o uso da estratégia de decomposição de classe *One Vs All* com o modelo de redes neurais *perceptron* multicamadas para a detecção de intrusão no contexto de Internet das Coisas.

O restante do trabalho está organizado da seguinte forma: a seção 2 apresenta conceitos básicos de IoT, Fog e Cloud; a seção 3 apresenta conceitos Sistemas de Detecção de Intrusão; a seção 4 apresenta conceitos de Inteligencia Artificial; a seção 5 descreve trabalhos correlatos; a seção 6 descreve a abordagem proposta; a seção 7 descreve a realização dos experimentos; a seção 8 apresenta discussões dos resultados; a seção 9 apresenta a conclusão.

2. IoT, Fog e Cloud

Internet das Coisas consiste na integração de um conjunto de objetos físicos, como sensores, atuadores, nós inteligentes e aplicações incorporadas por meio de uma rede para a interação com estados internos ou ambientes externos. Essa arquitetura utiliza protocolos da Internet para a criação de um novo ecossistema de aplicativos, levando a conexão generalizada de pessoas, serviços, sensores e objetos [Conti et al. 2018, Oppitz and Tomsu 2018].

A Computação em Nuvem (*Cloud Computing*) é um modelo que fornece recursos de computação de forma conveniente e sob demanda. Esses recursos são altamente configuráveis, providos de maneira fácil e rápida. Dentre esses recursos estão capacidade de armazenamento, processamento e disponibilização de softwares [Foster et al. 2008].

A Computação em Nevoeiro (*Fog Computing*) fica localizada próximo a borda da rede, resolvendo diversos problemas de aplicações IoT ligadas diretamente em serviços em Nuvem, possibilitando um novo ecossistema de aplicações [Bonomi et al. 2012]. Este é um conceito, introduzido pela Cisco em 2014, que distribui as funções de armazenamento, processamento, controle e rede para mais próximo do cliente [Chen et al. 2018].

3. Sistema de Detecção de Intrusão

Sistemas de Detecção de Intrusão (*Intrusion Detection System - IDS*) são ferramentas de segurança [Garcia-Teodoro et al. 2009] que tem como objetivo defender um sistema, executando contramedidas ou gerando alertas para uma entidade capaz de realizar ações apropriadas, quando o sistema foi comprometido [Axelsson 2000].

Dependendo do tipo de análise realizada, o IDS pode ser classificado em detecção por anomalia ou por assinatura. Na detecção por anomalia o IDS define todo o comportamento padrão e sinaliza todo o comportamento anormal a este padrão definido. Na detecção por assinatura o IDS compara as ações monitoradas com assinaturas definidas anteriormente no sistema. No primeiro, não é necessário ter conhecimento a priori das assinaturas dos ataques como no segundo, mas é preciso estimar o comportamento normal das entidades do sistema. Os dois tipos são semelhantes em termos de operação, mas a detecção baseada em assinatura tende a ser eficaz apenas em ataques conhecidos, em contrapartida detecção baseada em anomalia tem o potencial de detectar eventos intrusivos não visto anteriormente [Garcia-Teodoro et al. 2009, Axelsson 2000].

Um IDS também pode ser classificado pela origem da informação analisada. Alguns IDS capturam e analisam pacotes da rede para encontrar ataques, chamados de IDS baseados em redes ou *Network-based Intrusion Detection System* (NIDS), outros IDS analisam informações geradas pelo sistema operacional ou aplicações, chamados de IDS baseado em host ou *Host-based Intrusion Detection System* (HIDS) [Bace and Mell 2001].

Após a identificação da intrusão o IDS pode se comportar de forma ativa ou passiva. No comportamento passivo o IDS é programado para informar o evento detectado para o usuário responsável, através de mensagens consoles, *e-mail*, relatórios, entre outros. No comportamento ativo o IDS é capaz de ser programado pelo usuário a realizar ações pró-ativas e corretivas a um evento crítico identificado, como corrigir a vulnerabilidade do sistema, reconfiguração do *firewall*, entre outros [Jackson et al. 1999].

4. Aprendizado de Máquina

A Inteligência Artificial é um campo próspero com muitas aplicações práticas. Principalmente em tarefas que são resolvidas intuitivamente pelas pessoas, mas difíceis de serem descritas formalmente por um conjunto de regras matemáticas. Dentre elas estão o reconhecimento de palavras faladas e rostos em imagens [Goodfellow et al. 2016].

Aprendizado de máquina é um subconjunto da Inteligência Artificial que constrói um modelo matemático com base em dados e amostras a fim de fazer previsões ou decisões sem serem explicitamente programadas para realizar tal tarefa [Zhang 2020].

4.1. Redes Neurais Artificiais

Segundo [Anderson and McNeill 1992] as Redes Neurais Artificiais são modelos eletrônicos baseados na estrutura neural do cérebro. Esses modelos foram inspirados na biologia para resolver problemas com os quais a computação tradicional tem dificuldade em lidar. Eles também trazem um conjunto de palavras incomuns para a computação tradicional como comportar, aprender, generalizar, auto-organizar e esquecer.

Segundo [Chua and Yang 1988] as redes neurais são propostas para resolver problemas em diversas áreas, como otimização, programação linear e não linear, reconhecimento de padrões e visão computacional.

Atualmente aplicações que utilizam redes neurais estão cada vez mais populares. Uma grande vantagem de usar esses modelos é a capacidade de lidar com sistemas naturais complexos que possuem grandes quantidades de informações [Abiodun et al. 2018].

Redes neurais da classe *Feed-Forward*, são organizadas em camadas, onde os neurônios da camada anterior são conectados aos neurônios da camada seguinte, mas não vice-versa. As redes neurais *Feed-Forward* não possuem ligações entre neurônios da mesma camada e não possuem ciclos em seus grafos de conexão. Redes que possuem ciclos em seu grafo de conexão são chamadas de Redes Neurais Recorrentes, onde a saída de um neurônio pode servir de entrada para o mesmo, ou para neurônios de camadas anteriores. As redes neurais *Feed-Forward* possuem uma camada de entrada e uma camada de saída. Elas podem possuir camadas ocultas, localizadas entre a camada de entrada e saída. Se a rede possuir mais de uma camada oculta ela é chamada de redes neurais profundas (*deep learning*) [Haykin et al. 2009].

4.2. Estratégia de decomposição de problemas multiclasse

A classificação multiclasse mapeia um espaço de recurso de entrada com K classes de saídas, $K > 2$. No entanto, muitos classificadores funcionam melhor para problemas de duas classes de saída ($K = 2$). Assim uma abordagem é decompor o problema em K problemas de duas classes. Várias abordagens foram propostas, as mais populares são a *One vs All* (OVA) e *One Vs One* (OVO) [Oong and Isa 2012]

Na abordagem *One vs All* um problema de K classes é decomposto em K problemas onde K classificadores são criados para discriminar sua respectiva classe em relação a todas as outras.

Na abordagem *One vs One* um problema de K classes é decomposto em pares formando $K(K-1)/2$ problemas, onde cada classificador é responsável por distinguir a classe i da classe j.

5. Trabalhos correlatos

Nesta seção serão apresentadas soluções propostas por outros autores na utilização de redes neurais para detecção de intrusão. Os artigos escolhidos apresentam aspectos relacionados com este trabalho em relação a detecção de intrusão utilizando técnicas de redes neurais.

Os autores [Hodo et al. 2016] propuseram uma rede neural para detecção de intrusão de um sistema IoT. Os autores afirmaram que a rede neural criada obteve 99,4% de

precisão. Os autores não especificaram quais informações estavam sendo utilizadas para fazer a análise do tráfego da rede, dificultando a reprodução dos testes.

Os autores [Le et al. 2019] propuseram uma nova estrutura de IDS para melhorar a precisão de classificação de ataques do tipo *Remote-to-Local* (R2L) e *User-to-Root* (U2R). Os autores utilizaram os conjuntos de dados NSL-KDD de 2010 e o ISCX de 2012. Os autores compararam a proposta deles com as de outros autores e o modelo dos autores apresentaram uma precisão superior nos dois conjuntos de dados.

Os autores [Kanimozhi and Jacob 2019] propuseram utilizar redes neurais para detecção de intrusão de ataques *Botnet*. Os autores utilizaram o conjunto de dados CSE-CIC-IDS2018 [Sharafaldin et al. 2018]. A rede neural foi treinada com 1048575 registros que possuíam 80 atributos utilizando a técnica de amostragem *Cross Validation* com 10 *Fold*. O modelo criado obteve a área abaixo da curva ROC igual 1 e uma precisão próxima a 0.99.

Os autores [Majumder et al. 2020] propuseram um modelo híbrido de Redes Neurais Artificial (ANN) e Redes Neurais *Spiking (SNN)* para detecção de intrusão. Os autores usaram o conjunto de dados NSL-KDD. Os autores testaram 3 cenários. O primeiro cenário um modelo ANN que obteve 77,83% de precisão. O segundo cenário o modelo SNN que obteve 79,31% de precisão. O terceiro cenário o modelo híbrido ANN-SNN proposto que obteve 85,73% de precisão.

Os autores [Sun et al. 2020] propuseram o modelo *Multi-Task Learning* (MTL) para resolver o problema de detecção de intrusão multiclasse. Foram utilizados dois conjuntos de dados, UNSW-NB15 e CSE-CIC-IDS-2018. Foi utilizado o método *SMOTH* para sobreamostrar as amostras minoritárias. Para comparação, foram realizados experimentos com os modelos SVM, DT, KNN, AdaBoost e BSPL-GN. Resultados experimentais mostram que o método propostos pelos autores têm o melhor desempenho.

Ao longo desta seção foram descritos 5 trabalhos correlatos. São diferentes propostas que usaram métodos de aprendizado de máquina para buscar melhorar a detecção de instrução em rede. Destes trabalhos, 2 deles propuseram métodos de detecção binária e 3 deles a detecção multiclasse. A detecção multiclasse é importante, pois fornecem informações mais precisas para o responsável da rede tomar contramedidas eficazes. Em contrapartida é mais difícil obter uma alta taxa de precisão na discriminação das diferentes classes. A área de aprendizado de máquina está se desenvolvendo muito nos últimos anos e tem muitos métodos que podem ser explorados para detecção multiclasse.

6. Abordagem Proposta

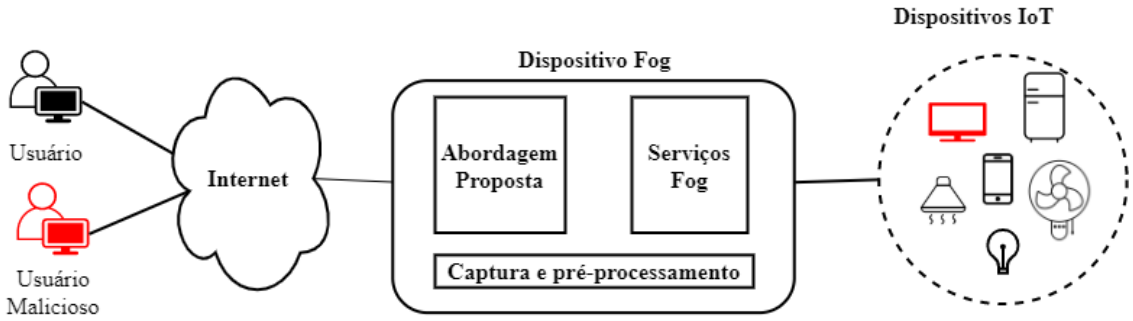
Considerando o crescente aumento de sistemas IoT, suas deficiências em relação a segurança e as pesquisas relacionadas sobre a utilização de redes neurais para detecção de intrusão, este trabalho propõe investigar a utilização da estratégia de decomposição de problemas para melhorar o desempenho de redes neurais na classificação multiclasse no problema de detecção de intrusão.

6.1. Contextualização da Proposta

O trabalho considera o contexto de um sistema IoT que utiliza o dispositivo *fog* para prover serviços e dados para os usuários. Neste contexto, usuários maliciosos podem

tentar invadir ou corromper o sistema para roubar dados dos usuários. A Figura 1 mostra a arquitetura desse sistema, onde a abordagem proposta é inserida no dispositivo *fog* para detectar a intrusão. Este trabalho considera que o dispositivo possui mecanismos para captura e pré-processamento do tráfego TCP/IP originados da rede, e além disso, prover os dados necessários para abordagem proposta analisar se está ocorrendo um ataque na rede e que tipo de ataque é esse.

Figure 1. Visão geral da abordagem proposta.



Fonte: O autor.

A seguir, são apresentados os principais aspectos da abordagem proposta.

6.1.1. Descrição da Proposta

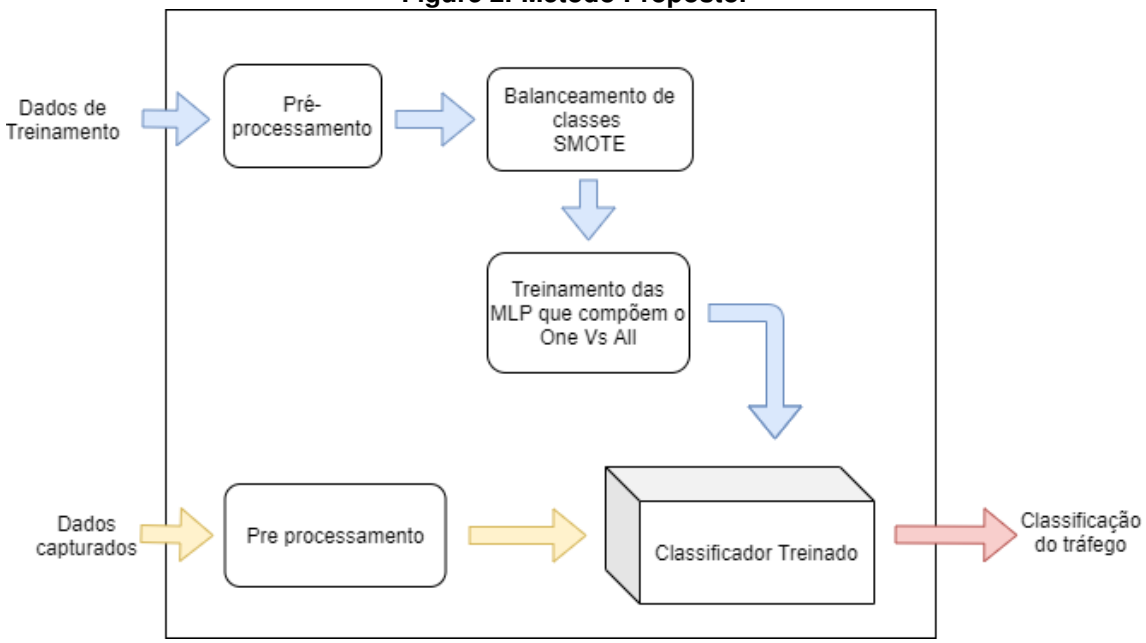
Nesta seção são fornecidos maiores detalhes sobre a abordagem proposta para detecção e identificação de intrusões. A Figura 2 ilustra o método proposto neste trabalho. A abordagem necessita de um conjunto de dados para a realização do treinamento do modelos de classificação. Os dados selecionados para o treinamento são formatados e pré-processados para retirada de registros inválidos. Devido a disparidade de proporção de tráfego entre os tipos de ataques, a abordagem conta com uma etapa de balanceamento para que os modelos tenham dados suficientes para compreender os padrões dos ataques que possuem pouca ocorrência. Por fim, é realizado o treinamento da abordagem de classificação gerando um modelo final capaz de classificar novos dados.

Após o processo de treinamento o modelo estará pronto para operar recebendo novos dados capturados da rede para serem analisados e classificados. Durante a execução real os dados são capturados, pré-processados e submetidos para o classificador treinado. O classificador então realiza a detecção e a identificação da categoria do tráfego.

6.1.2. Classificador

Neste trabalho é investigada a estratégia de decomposição de classes *One Vs All*. Nessa estratégia K classificadores são criados para K classes, onde cada classificador é treinado para discriminar sua respectiva classe [Oong and Isa 2012]. A saída de cada classe é passada para uma função de combinação multiclasse que retorna a saída da rede. A Figura 3 ilustra a abordagem baseada em OVA composta pelos modelos neurais.

Figure 2. Método Proposto.



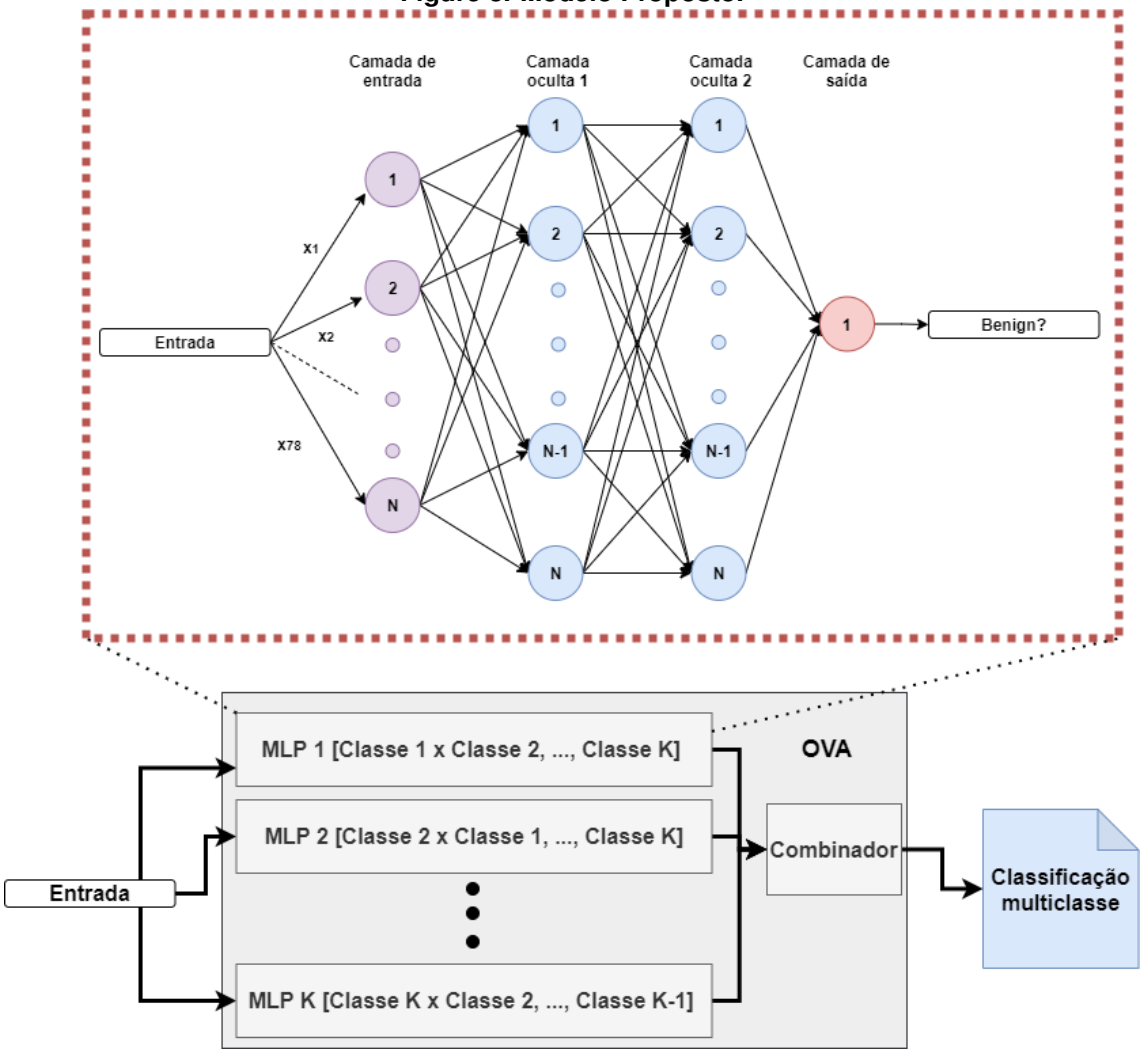
Fonte: O autor.

A abordagem proposta utiliza modelos neurais perceptron de multicamadas. Conforme pode ser observado na Figura 3, a camada de entrada, assim como, as suas duas camadas ocultas, possuem n neurônios, onde n corresponde a quantidade de atributos do tráfego considerado. Os neurônios das camadas ocultas, responsáveis pelo aprendizado da rede, utilizam a função de ativação *ReLU* [Agarap 2019]. Esta função de ativação retorna o valor recebido pelo somador se o mesmo for maior que zero, se os valores forem negativos retorna zero. Por ser uma função simples, tornou-se uma função muito utilizada em diversos tipos de redes neurais, pois é mais fácil de treinar e normalmente atingem bom resultados [Agarap 2019]. A camada de saída é composta por apenas 1 neurônio que utiliza a função de ativação *sigmoide*.

Conforme mencionado anteriormente, o método proposto utiliza K modelos neurais, onde cada um vai discriminar se a entrada pertence ou não a sua respectiva classe. Cada modelo neural é treinado para discriminar sua respectiva classe em relação a todas as outras. A entrada é encaminhada para cada MLP e a saída de cada rede entra em um combinador que classifica a entrada com sendo da classe que teve a rede neural com o maior valor de saída.

Além disso, a abordagem realiza um processo de balanceamento de dados antes do treinamento dos modelos neurais. Esta etapa é realizada com o objetivo de contornar à dificuldade de se trabalhar com dados desbalanceados, onde algumas classes possuem uma quantidade muito superior de registros em relação as demais. Isto faz com que a rede neural tenha um desempenho melhor em classificar as classes com mais registros e classes com poucos registros acabam tendo baixo desempenho da rede. Por isso foi utilizada a técnica de *Synthetic Minority Oversampling Technique* (*SMOTE*) [Bowyer et al. 2011], que consiste em sintetizar os registros falsos para classes que contém poucos registros. Neste trabalho optou-se por sintetizar novos registros em todas as classes de instrução até

Figure 3. Modelo Proposto.



Fonte: O autor.

que elas alcancem a marca de 100.000 registros.

7. Avaliação

Nesta seção são descritos alguns aspectos da metodologia utilizada para avaliar a abordagem proposta. Em seguida são apresentados e discutidos os resultados alcançados através dos experimentos.

7.1. Experimentos

A metodologia utilizada para realização dos experimentos é descrita a seguir. Inicialmente são apresentadas as características da base de dados e a sua divisão em conjuntos para treino e teste. Em seguida são apresentadas as métricas de classificação consideradas na avaliação. Por fim, são descritos os experimentos realizados.

7.1.1. Base de dados

Neste trabalho foi utilizada a base de dados CSE-CIC-IDS2018[1] [Sharafaldin et al. 2018] para realizar os experimentos. A base foi concebida através de um projeto colaborativo entre o *Communications Security Establishment* (CSE) e o *Canadian Institute for Cybersecurity* (CIC). O projeto tem como objetivo desenvolver uma abordagem sistemática para gerar conjunto de dados de instrução em rede diversificado, baseado em perfis de usuários que contém um conjunto de eventos combinado com comportamentos visto na rede.

O conjunto de dados contém os seguintes ataques:

- **Ataque de força bruta:** Ataque que tenta adivinhar a senha de um usuário em sites, usando dicionários específicos com senhas mais usadas e ferramentas para automatizar o processo.
- ***Botnet*:** O atacante usa *malware* do tipo cavalo de troia para tentar controlar as máquinas das vítima.
- **Negação de serviço:** O atacante tenta ocupar todos os recursos do servidor enviado várias requisições para o servidor responder, deixando o serviço indisponível para os usuários.
- **Negação de serviço distribuída:** O atacante usa diversas máquinas para realizar o ataque de negação de serviço.
- **Ataque na web:** São um conjunto de diferentes ataques visando explorar vulnerabilidade de sites *web*, incluindo injeção SQL, injeção de comando e *upload* irrestrito de arquivo.
- **Infiltração na rede:** Neste cenário foi enviado arquivo malicioso para a vítima, que ao ser executado realizou uma varredura na rede interna visando buscar outras vulnerabilidades e explorá-las.

7.1.2. Pré-processamento da base de dados

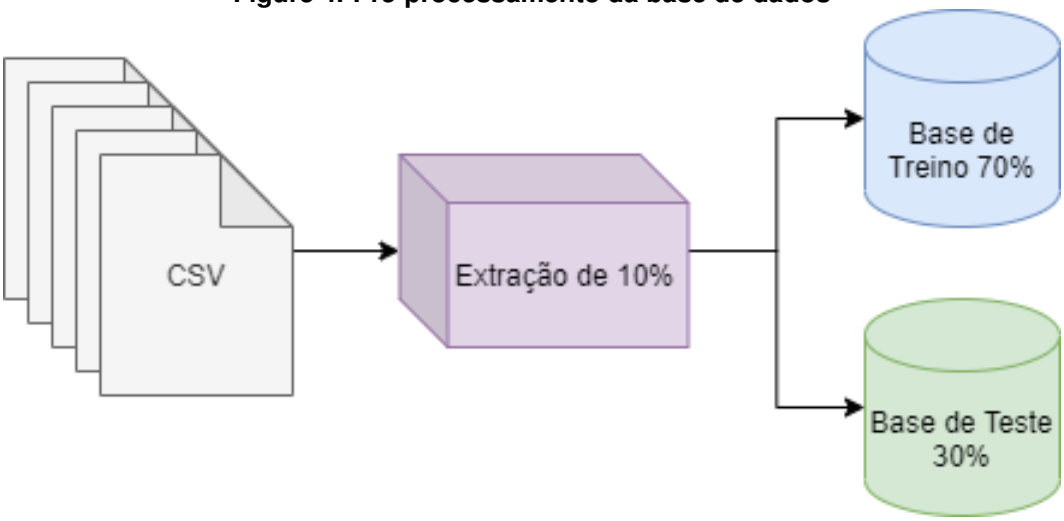
A base CSE-CIC-IDS-2018 possui um grande conjunto de dados divididos em 10 arquivos do tipo csv totalizando aproximadamente 6,5 Gb de dados. Devido a enorme quantidade de dados e a dificuldade de recursos para trabalhar com a base completa ouve a necessidade de utilizar um subconjunto de 10% da base para realizar os experimentos. Sendo que o simples fato de realizar o carregamento da base no plataforma, esgotava a quantidade de memória RAM disponível. Essa extração de 10% dos registros foi realizada de forma aleatória, mas mantendo a proporção da quantidade de registros por classes. A Figura 4 apresenta um diagrama ilustrando o processo.

Inicialmente cada arquivo csv passou por um pré-processamento. Nesta etapa foram removidos alguns atributos que apenas alguns arquivos continham, sendo: Flow ID, Src IP, Src Port, Dst IP. Em seguida também foi realizada a remoção de registros inválidos. Foram removidos os registros com valores nan e inf.

Além disso, foi realizado o agrupamento das classes por tipo de ataque. Originalmente os registros eram classificados pelo tipo e ferramenta de ataque que foi utilizada. A Tabela 1 mostra como foi feito o novo agrupamento. Posteriormente, as novas classes

[1] https://registry.opendata.aws/cse-cic-ids2018/

Figure 4. Pré-processamento da base de dados



Fonte: O autor.

foram transformadas para categorias numéricas, de modo, a tornar possível o treinamento dos modelos neurais.

Table 1. Agrupamento das classes

Classes Agrupadas	**Classes Originais**
Benign	*Benign*
Bot	*Bot*
Brute Force	*SSH-Bruteforce*
	FTP-BruteForce
Web Attack	*Brute Force -Web*
	Brute Force -XSS
	SQL Injection
DDOS	*DDOS attack-HOIC*
	DDOS attack-LOIC-UDP
	DDoS attacks-LOIC-HTTP
DoS	*DoS attacks-GoldenEye*
	DoS attacks-Slowloris
	DoS attacks-Hulk
	DoS attacks-SlowHTTPTest
Infiltration	*Infiltration*

Outro processo realizado no pré-processamento da base foi a normalização dos dados. A padronização do conjunto de dados ajuda a melhorar alguns classificadores baseados em aprendizado de maquina que precisam que seus recursos individuais estejam normalmente distribuído. O método utilizado para a normalização dos dados foi o escalonamento padrão, dado pela Equação 1, sendo x a amostra, u a média e s o desvio padrão. A média e o desvio padrão são obtidos com base na estatística de cada atributo (coluna) do conjunto de dados.

$$z = \frac{x - u}{s} \tag{1}$$

Em seguida foi utilizado a técnica de *Hold Out*, onde há uma divisão de dados entre treino e teste. Dessa forma, foram usados 70% da base para treinamento e 30% para o teste.

7.1.3. Métricas de Avaliação

Para avaliação do modelo serão utilizadas algumas métricas de avaliação comumente empregadas para avaliar aplicações de classificação. Inicialmente, a partir dos experimentos são obtidas as seguintes medidas :

- **Verdadeiro Positivo (VP):** Instâncias classificadas corretamente como intrusão pelo método de detecção.
- **Verdadeiro Negativo (VN):** Instâncias classificadas corretamente como normal pelo método de detecção.
- **Falso Positivo (FP):** Instâncias classificadas como intrusão pelo método de detecção, mas que na realidade são instâncias normais.
- **Falso Negativo (FN):** Instâncias classificadas como normal pelo método de detecção, mas que na realidade são instâncias de intrusão.

A partir das medidas supracitadas é possível calcular seguintes métricas de avaliação [Almiani et al. 2020]:

- **Acurácia (ACC):** Proporção de instâncias classificadas corretamente sobre o número total de classificação. Ela é calculada através da Equação 2.

$$ACC = \frac{VP + FP}{VP + VN + FP + FN} \tag{2}$$

- **Precisão:** Apresenta a taxa de instâncias classificadas corretamente como intrusão sobre o total de instâncias classificadas intrusão. Ela pode ser obtida através da Fórmula 3.

$$PRECISAO = \frac{VP}{VP + FP} \tag{3}$$

- **Sensibilidade (*Recall*):** Apresenta a taxa de instâncias classificadas como intrusão sobre o total de instâncias intrusão. Ela pode ser calculada através da Fórmula 4.

$$RECALL = \frac{VP}{VP + FN} \tag{4}$$

- **Especificidade (*True Negative Rate - TNR*) :** Proporção classificada corretamente instancias normais sobre o total de instâncias normais. Ela pode ser calculada através da Fórmula 5.

$$TNR = \frac{FP}{FP + FV} \tag{5}$$

• ***F1-score*:** Esta pontuação é para avaliar a qualidade geral do modelo, calculando a média harmônica entre a precisão e o recall. Ela pode ser obtida através da Formula 6.

$$f1\_score = 2 * \frac{PRECISAO * RECALL}{PRECISAO + RECALL} \tag{6}$$

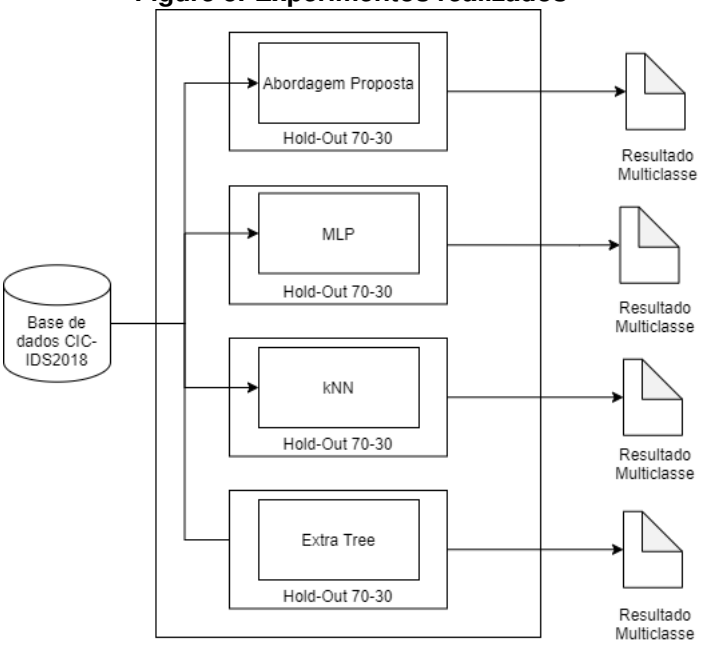
• **Acurácia balanceada:** Métrica utilizada em classificação multiclasse onde os dados são desbalanceados, onde a acurácia da classe é ponderada pelo número de instâncias. Pode ser obtido pela Fórmula 7.

$$ACC\_BAL = \frac{RECALL + TNR}{2} \tag{7}$$

7.1.4. Experimentos Realizados

Para fins de comparação e avaliação da abordagem proposta, foram realizados experimentos com a solução proposta e com outras 3 abordagens de classificação. Sendo eles um modelo MLP, um modelo *ExtraTree* (ET) e uma *K-Nearest Neighbor* (kNN). A Figura 5 mostra os experimentos realizados com os 4 modelos. Após o pré-processamento descrito na seção 7.1.2, foi realizado o treinamento dos 4 modelos com o conjunto de teste.

Figure 5. Experimentos realizados



Fonte: O autor.

A *Multilayer Perceptron* (MLP) utilizada para comparação é similar ao modelo neural que compõe a solução proposta. A camada de entrada, assim como, as suas duas camadas ocultas, possuem 78 neurônios com função de ativação *relu*. No entanto, a camada de saída possui 7 neurônios com função de ativação *softmax*, um para cada classe, onde cada neurônio indica se a entrada pertence ou não a respectiva classe. A entrada vai pertencer a classe do neurônio que possui o maior valor.

O *K—Nearest Neighbors* (KNN) é um algoritmo muito utilizado na área de aprendizado de máquina de mineração de dados (*data mining*). A classificação é realizada através da comparação de similaridade em relação aos dados de treinamento. A classe majoritária entre os K registros mais similares é atribuída ao dado a ser classificado. Nesse experimento, utilizou-se K com valor 1.

Por fim, a *Extra Trees* (ET) consiste na criação de várias árvores de decisões aleatórias. Em um problema de classificação cada Árvore de Decisão vai classificar a entrada, votando em uma classe. A classe que receber mais votos será a saída do modelo.

7.1.5. Materiais utilizados

Neste trabalho foi utilizado a plataforma do *Google Colaboratory* para realização dos experimentos. É uma plataforma em nuvem que disponibiliza um ambiente de execução em *python*. A máquina possui o processador AMD EPYC 7B12 com 13GB de memória RAM. O sistema operacional é Ubuntu 18.4 LTS.

Os experimentos foram realizados na linguagem *python* versão 3.7. Foram utilizadas as seguintes bibliotecas:

- *Pandas*: é uma biblioteca para manipulação e analise de dados, que foi utilizada para carregar a base de dados e realizar processamento dos dados.
- *Scikit-learn*: é uma biblioteca que possui muitas ferramentas para auxiliar o processo de criação e treinamento de modelos de aprendizado de maquina. Foi utilizada para implementações dos modelos treinandos, funções de separação dos dados entre treino e teste, além da função para aplicação da técnica SMOTE na base de dados.

7.2. Resultados

Os resultados das quatro abordagens foram obtidos utilizando o conjunto de teste que contém registros que não foram usados pelos modelos na fase de treinamento. Cada modelo descrito na Figura 5 foi testado e os resultados obtidos foram utilizados para calcular as métricas descritas na Seção 7.1.3.

A Tabela 2 apresenta o resultado dos experimentos com a abordagem proposta, são apresentados os resultados de cada classe e a avaliação geral do modelo. Os resultados obtidos mostram-se satisfatórios. A abordagem alcançou *recall* de 0,99 para o tráfego normal. A métrica *recall* indica a porcentagem de tráfego normal que foi identificado como benigno dentre todos os registros de tráfego normal presentes no conjunto de teste. Além disso, apresentou altas taxas de *recall* para a detecção e identificação de ataques *Botnet*, DDoS e *Brute Force*, nos três casos apresentou *recall* superior a 0,95. Para os ataques DoS, a abordagem apresentou uma boa taxa de detecção, aproximadamente 0,89, conforme pode ser observado na Tabela 2. No entanto, a abordagem não foi capaz de alcançar uma boa taxa de detecção para a classe de ataques *Infiltration*, apresentando um *recall* de apenas 0,23. Para a classe *Web Attack* a abordagem foi capaz de alcançar uma taxa de *recall* de 0,93. No entanto, apresentou uma precisão baixa, e este fato refletiu na baixa taxa de *F1-score*, métrica que consiste em uma média harmônica entre precisão e *recall*.

Table 2. Resultados do modelo proposto OVA.

Classe	*acc*	*loss*	*precision*	*recall*	*TNR*	*f1-score*
Benign	0,99	0,01	0,99	0,99	0,95	0,99
Bot	1	0	1	1	1	1
Brute Force	0,99	0,01	0,83	0,95	1	0,89
DDOS	1	0	1	1	1	1
DoS	0,99	0,01	0,97	0,89	1	0,92
Infiltration	0,99	0,01	0,31	0,23	0,99	0,27
Web Attack	1	0	0,08	0,93	1	0,15
Acurácia				**0,98**		
Acurácia Balanceada				**0,85**		
Precisão				**0,98**		

No geral, a abordagem proposta apresentou uma acurácia extremamente alta, o motivo se deve principalmente ao fato de ter apresentado dificuldades na detecção apenas das classes que possuíam poucos registros, as classes *Infiltration* e *Web Attack*. Desse modo, a taxa de acurácia geral não foi prejudicada pelo desempenho ruim em relação a estas classes. A acurácia balanceada, apresentada também na Tabela 2, permite mensurar de modo mais realístico o desempenho. A abordagem proposta alcançou 0,85 de acurácia balanceada.

Na Tabela 3 são apresentados os resultados alcançados pelo modelo MLP único no experimento.

Table 3. Resultados do modelo MLP.

Classe	acc	loss	precision	recall	TNR	f1-score
Benign	0,99	0,01	0,99	1	0,94	0,99
Bot	1	0	1	1	1	1
Brute Force	0,99	0,01	0,84	0,94	1	0,89
DDOS	1	0	1	1	1	1
DoS	0,99	0,01	0,96	0,89	1	0,93
Infiltration	0,99	0,01	0,5	0,02	1	0,04
Web Attack	1	0	0,8	0,43	1	0,56
Acurácia				**0,98**		
Acurácia Balanceada				**0,75**		
Precisão				**0,98**		

Assim como a abordagem proposta, o modelo MLP foi capaz de obter ótimas taxas de *recall* na identificação de tráfego normal e de ataques *Botnet*, *Brute Force*, DDoS e DoS. O modelo teve problemas com as classes *Infiltration* e *Web Attack*, apresentando *recall* de 0,02 e 0,43, e *F1-score* de 0,04 e 0,56, respectivamente. De modo geral, o modelo neural foi capaz de obter uma acurácia balanceada de 0,75.

Os resultados obtidos pelo algoritmo kNN são apresentados na Tabela 4. Assim como as abordagens anteriores, o algoritmo apresentou resultados positivos para a maioria das classes. Ele também encontrou dificuldades para identificar ataques *Infiltration* e *Web*

Attack, obtendo um *F1-score* de 0,14 e 0,67 respectivamente. No geral, esta abordagem alcançou uma acurácia de 0,78, superando o modelo neural. No entanto, cabe destacar, que este algoritmo realiza muitas comparações em tempo de detecção, o que pode levar a um *delay* maior de detecção.

Table 4. Resultados do modelo kNN.

Classe	*acc*	*loss*	*precision*	*recall*	*TNR*	*f1-score*
Benign	0,98	0,02	0,99	0,99	0,95	0,99
Bot	1	0	1	1	1	1
Brute Force	0,99	0,01	0,99	0,75	1	0,85
DDOS	1	0	1	1	1	1
DoS	0,99	0,01	0,87	1	0,99	0,93
Infiltration	0,98	0,02	0,14	0,13	0,99	0,14
Web Attack	1	0	0,74	0,61	1	0,67
Acurácia				**0,98**		
Acurácia Balanceada				**0,78**		
Precisão				**0,98**		

Por fim, a Tabela 5 apresenta os resultados do experimento efetuado com o modelo *Extra Tree*. Assim como as demais abordagens, ele alcançou ótimas taxas de *recall* para os ataques *Botnet*, *Brute Force*, DDoS e DoS. As principais dificuldades também estão relacionadas com com os ataques *Infiltration* e *Web Attack* com *recall* de 0,10 e 0,50, respectivamente. A acurácia balanceada apresentada pela Extra Tree foi de 0,78.

Table 5. Resultados do modelo ET.

Classe	*acc*	*loss*	*precision*	*recall*	*TNR*	*f1-score*
Benign	0,99	0,01	0,99	1	0,95	0,99
Bot	1	0	1	1	1	1
Brute Force	0,99	0,01	0,84	0,94	1	0,89
DDOS	1	0	1	1	1	1
DoS	0,99	0,01	0,96	0,89	1	0,93
Infiltration	0,99	0,01	0,21	0,1	1	0,14
Web Attack	1	0	0,93	0,5	1	0,65
Acurácia				**0,98**		
Acurácia Balanceada				**0,78**		
Precisão				**0,98**		

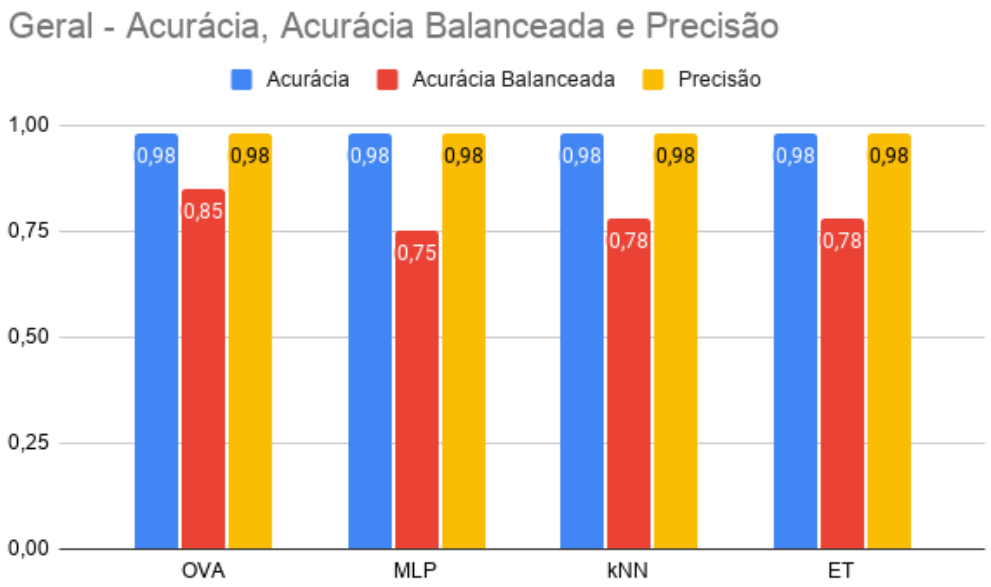
8. Discussões

Nesta seção, serão discutidos os resultados e comparados os desempenhos dos diferentes modelos. A Figura 6 apresenta um gráfico comparando o desempenho geral dos modelos avaliados.

A acurácia alta presente nos resultados das abordagens se justifica pelo fato delas terem alcançado um ótimo desempenho na detecção do tráfego normal que possui uma grande quantidade de registros. No entanto, ao se analisar a acurácia balanceada,

métrica que pondera as classes através da proporção de tráfego, é possível observar que este número tem uma queda. As abordagens MLP, kNN e *Extra Tree* obtiveram 0,75, 0,78 e 0,78 de acurácia balanceada, respectivamente. A abordagem proposta foi capaz de melhorar essa métrica, alcançando uma acurácia balanceada de 0,85. Portanto, nas métricas gerais, a abordagem foi capaz de obter um desempenho interessante, superando as demais.

Figure 6. Gráfico comparação do desempenho geral dos modelos.



Fonte: O autor.

A Tabela 6 mostra uma comparação dos resultados da abordagem proposta e do trabalho dos autores [Sun et al. 2020].

Table 6. Comparação com trabalho do estado da arte.

	Modelo Proposto (OVA)			**[Sun et al. 2020]**		
	PRE	**Recall**	**F1-Score**	**PRE**	**Recall**	**F1-Score**
Classe Benign	**0,99**	**0,99**	**0,99**	-	-	-
Classe Bot	**1,00**	**1,00**	**1,00**	0,53	0,88	66,42
Classe Brute Force	**0,83**	**0,95**	**0,89**	0,58	0,94	71,49
Classe DDOS	**1,00**	**1,00**	**1,00**	0,97	**1,00**	98,17
Classe DoS	**0,97**	**0,89**	**0,92**	0,90	0,88	88,95
Classe Infiltration	0,31	0,23	0,27	**0,54**	**0,74**	**62,13**
Classe Web Attack	0,08	**0,93**	0,15	**0,56**	0,90	**68,96**

É possível observar que a abordagem proposta neste trabalho foi capaz de obter ótimos resultados e superar a outra abordagem em todas as métricas para as classes *Botnet*, *Brute Force*, *DDoS* e *DoS*. Por poutro lado, em relação a classe de ataques *Infiltration*, o desempenho da proposta de [Sun et al. 2020] alcançou melhores resultados. Em relação a classe *Web Attacck*, a abordagem proposta alcançou o maior *recall*, porém teve uma

baixa precisão, indicando taxa de falsos positivos alta. Cabe destacar aqui que o trabalho de [Sun et al. 2020] não apresentou os resultados para a identificação de tráfego normal. Métricas baixas nesse quesito podem indicar alta taxa de bloqueio de tráfego normal, que não deveria ser bloqueado, portanto, é uma métrica muito importante também.

No geral, a abordagem proposta neste trabalho apresentou um bom resultado. Alcançando excelentes taxas de detecção na maioria dos ataques avaliados. No entanto, não foi capaz de apresentar melhoras no desempenho de detecção de ataques como *Infiltration* e *Web Attack*. Portanto, são necessários maiores estudos na aplicação de técnicas de aprendizado de máquina na detecção e classificação de intrusão em redes IoT.

9. Conclusão

Os sistemas IoT estão cada vez mais comuns e presentes na vida das pessoas, podendo ser aplicados em diversas áreas e diferentes contextos. Entretanto, por causa da baixa capacidade de processamento e memória apresentam uma séries de preocupações de segurança. Este trabalho teve como objetivo propor uma abordagem para detecção e identificação de intrusões baseado no método *One Vs All* e em modelos neurais MLP. Além disso, a abordagem contou com a técnica SMOTE para criar novos registros para balancear a base.

A avaliação da abordagem proposta foi realizada através da técnica *Hold Out* e com o conjunto de dados CSE-CIC-IDS2018, sendo um grande conjunto de dados recentes que vem sendo utilizando em pesquisas nessa área. Foram realizados experimentos para avaliar a abordagem proposta em relação as técnicas de aprendizado de máquina. No geral todos os modelos tiveram um bom desempenho, destacando o modelo proposto que obteve uma acurácia balanceada de 0,85% sendo a maior em comparação aos modelos. No entanto, todas as abordagens tiveram dificuldade em identificar os ataques *Infiltration* e *Web Attack*. Como trabalhos futuros, é importante destacar que são necessários maiores estudos na aplicação de técnicas de aprendizado de maquina na detecção e classificação de intrusão em redes IoT.

References

Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., and Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11):e00938.

Agarap, A. F. (2019). Deep learning using rectified linear units (relu).

Almiani, M., AbuGhazleh, A., Al-Rahayfeh, A., and Razaque, A. (2020). Cascaded hybrid intrusion detection model based on som and rbf neural networks. *Concurrency and Computation: Practice and Experience*, 32(21):e5233.

Anderson, D. and McNeill, G. (1992). Artificial neural networks technology. *Kaman Sciences Corporation*, 258(6):1–83.

Axelsson, S. (2000). Intrusion detection systems: A survey and taxonomy. Technical report, Technical report.

Bace, R. and Mell, P. (2001). Nist special publication on intrusion detection systems. Technical report, BOOZ-ALLEN AND HAMILTON INC MCLEAN VA.

Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012). Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16.

Bowyer, K. W., Chawla, N. V., Hall, L. O., and Kegelmeyer, W. P. (2011). SMOTE: synthetic minority over-sampling technique. *CoRR*, abs/1106.1813.

Chen, Y.-D., Azhari, M. Z., and Leu, J.-S. (2018). Design and implementation of a power consumption management system for smart home over fog-cloud computing. In *2018 3rd International Conference on Intelligent Green Building and Smart Grid (IGBSG)*, pages 1–5. IEEE.

Chua, L. O. and Yang, L. (1988). Cellular neural networks: Theory. *IEEE Transactions on circuits and systems*, 35(10):1257–1272.

Conti, M., Dehghantanha, A., Franke, K., and Watson, S. (2018). Internet of things security and forensics: Challenges and opportunities.

Foster, I., Zhao, Y., Raicu, I., and Lu, S. (2008). Cloud computing and grid computing 360-degree compared. In *2008 grid computing environments workshop*, pages 1–10. Ieee.

Frustaci, M., Pace, P., Aloi, G., and Fortino, G. (2017). Evaluating critical security issues of the iot world: Present and future challenges. *IEEE Internet of things journal*, 5(4):2483–2495.

Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., and Vázquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. *computers & security*, 28(1-2):18–28.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*, volume 1. MIT press Cambridge.

Haykin, S. et al. (2009). Neural networks and learning machines. *Upper Saddle River: Pearson Education*, 3.

Hodo, E., Bellekens, X., Hamilton, A., Dubouilh, P.-L., Iorkyase, E., Tachtatzis, C., and Atkinson, R. (2016). Threat analysis of iot networks using artificial neural network intrusion detection system. In *2016 International Symposium on Networks, Computers and Communications (ISNCC)*, pages 1–6. IEEE.

Hughes, K., McLaughlin, K., and Sezer, S. (2020). Dynamic countermeasure knowledge for intrusion response systems. In *2020 31st Irish Signals and Systems Conference (ISSC)*, pages 1–6. IEEE.

Iorga, M., Feldman, L., Barton, R., and Martin, M. (2018). Fog computing conceptual model. special publication (nist sp)-500–325.

Jackson, K. A. et al. (1999). Intrusion detection system (ids) product survey. *Los Alamos National Laboratory*.

Kanimozhi, V. and Jacob, T. P. (2019). Artificial intelligence based network intrusion detection with hyper-parameter optimization tuning on the realistic cyber dataset cse-cic-ids2018 using cloud computing. In *2019 International Conference on Communication and Signal Processing (ICCSP)*, pages 0033–0036. IEEE.

Le, T.-T.-H., Kim, Y., Kim, H., et al. (2019). Network intrusion detection based on novel feature selection model and various recurrent neural networks. *Applied Sciences*, 9(7):1392.

Majumder, D., Singh, A., Ghosh, P., and Phadikar, S. (2020). A novel snn-ann based ids in cloud environment. In *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, pages 913–918. IEEE.

Oong, T. H. and Isa, N. A. M. (2012). One-against-all ensemble for multiclass pattern classification. *Applied Soft Computing*, 12(4):1303–1308.

Oppitz, M. and Tomsu, P. (2018). Internet of things. In *Inventing the Cloud Century*, pages 435–469. Springer.

Sharafaldin, I., Lashkari, A. H., and Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *ICISSp*, pages 108–116.

Soe, Y. N., Feng, Y., Santosa, P. I., Hartanto, R., and Sakurai, K. (2019). Implementing lightweight iot-ids on raspberry pi using correlation-based feature selection and its performance evaluation. In *International Conference on Advanced Information Networking and Applications*, pages 458–469. Springer.

Sun, L., Zhou, Y., Wang, Y., Zhu, C., and Zhang, W. (2020). The effective methods for intrusion detection with limited network attack data: Multi-task learning and oversampling. *IEEE Access*, 8:185384–185398.

Vikram, N., Harish, K., Nihaal, M., Umesh, R., and Kumar, S. A. A. (2017). A low cost home automation system using wi-fi based wireless sensor network incorporating internet of things (iot). In *2017 IEEE 7th International Advance Computing Conference (IACC)*, pages 174–178. IEEE.

Zhang, X.-D. (2020). Machine learning. In *A Matrix Algebra Approach to Artificial Intelligence*, pages 223–440. Springer.