Faculdade de ciência e Tecnologia da Universidade de Coimbra

Departamento de Engenharia Informática

Deteção de intrusões em ambientes IoT com CoAP

Dorivaldo Francisco da Silva

Relatório de Estágio no âmbito do Mestrado em Engenharia Informática, especialização em Comunicações Serviços e Infraestruturas, orientado pelo professor Doutor António Jorge da Costa Granjal e pelo Professor Doutor Nuno Lourenço, apresentada à Faculdade de Ciências e Tecnologia / Departamento de Engenharia Informática

Janeiro de 2019

UNIVERSIDADE D
COIMBRA

Agradecimentos

A Deus pelo dom da vida, saúde e graça para caminhar em direção a realização dos meus alvos.

A Universidade de Coimbra e em particular ao departamento de engenharia informática por me terem recebido de braços abertos e pela gama de conhecimento que colocaram a meu dispor, que me serão de grande utilidade para desafios futuros.

Aos meus orientadores professores Dr. Jorge Granjal e Nuno Lourenço, por todo apoio e suporte durante a realização do trabalho.

Aos meus pais João António da Silva Domingos e Celestina Natália Francisco por todo incentivo e suporte.

As minhas tias Lucrécia Castelo e Maria Castelo por todo amor, incentivo e apoio incondicional.

Aos meus amigos da universidade de Coimbra, Joel Pires, Hélder Sena, Jomar Domingos, Bongo Caisso, Ronaldo Júnior, Carlos Paiva, irmãos em Cristo Jesus e a todos aqueles que direta ou indiretamente contribuíram para que a realização deste trabalho fosse possível.

Resumo

A IoT (Internet of Things) é um conceito emergente ligado a conexão de WSN (Wireless Sensor Network), ou seja, dispositivos físicos com inteligência computacional como sensores e atuadores com a internet, permitindo que coisas sejam capazes de armazenar, processar e trocar informações entre si.

Neste novo paradigma vários pequenos dispositivos do nosso dia a dia com limitações a nível da capacidade de processamento, consumo de energia e armazenamento estão conectados com a internet e os mecanismos de segurança convencionais tornam-se ineficazes para estes ambientes, pois estes exigem um custo computacional elevado podendo provocar atrasos nas comunicações. Atendendo a falta de mecanismos de segurança para WSN, estudos têm sido realizados pela comunidade académica e de investigação para que possamos desfrutar de todas as valências da internet das coisas com segurança.

Usar a tecnologia IP (Internet Protocol) é uma grande vantagem para as WSN, visto que já existe uma infraestrutura amadurecida, com imensos protocolos abertos e diversas abordagens para sua gestão e segurança, mas estes protocolos são bastante complexos para o contexto de WSN, pelo que a IETF (Internet Engineering Task Force) desenvolveu mecanismos que permitem usar IPv6 (Internet Protocol Version 6) em redes de curto alcance e baixa potência de transmissão desenvolvendo novos protocolos para IoT.

Nesta pesquisa vamos considerar ambientes de internet das coisas com protocolos *Low-Power* 6LoWPAN (IPv6 Low-Power Wireless Personal Área Network) definido pelo RFC 4944, CoAP (Constrained Application Protocol) definido pelo RFC (Request for Comments) 7252, e o protocolo RLP (Routing Protocol for Low-Power and Loss Networks) definido pelo RFC 6550. Estes protocolos ajudam a integrar as WSN com a internet.

Por estarem conectados a internet as WSN ficam vulneráveis a ataques originados da internet, e a falta de mecanismos de segurança propícios para redes de sensores sem fio tem despertado o interesse da comunidade acadêmica e de investigação para a descoberta de técnicas que permitam a integração de WSN com a internet de maneira segura. Em muitas implementações soluções *standard* como os *firewall* são insuficientes para proteção dos sistemas, daí a necessidade de robustecer a segurança destes com IDS (Intrusion Detection System) para a deteção de ataques passivos ou ativos.

O resultado deste estudo culminará com o desenvolvimento de um IDS (Intrusion Detection System) baseado em anomalias para a IoT que funcione a nível da camada protocolar de aplicação CoAP (Constrained Aplication Protocol).

Implementaremos um classificador baseado em redes neuronais a semelhança de outro já existente baseado em SVM (Support Vector Machines), e o desempenho de ambos será comparado visto que pretendemos aprimorar as soluções para a problemática levantada.

As métricas que usaremos para avaliar o nosso IDS serão as usadas para um problema genérico de classificação binária, pois o sistema deverá ser capaz de avaliar se uma ação é ou não uma intrusão ao sistema

Palavras Chave: CoAP, 6LoWPAN, RLP, IoT, WSN, Aprendizagem de Máquina, IDS.

Abstract

The Internet of things is an emerging topic linked to data processing machine with computational intelligence, such as sensors and actuators on the internet. It allows things to be able to store, process, and exchange information with each other.

In this new paradigm several small devices of our daily life with limitations on the level of processing capacity, energy consumption and storage are connected to the internet and conventional security mechanisms become ineffective for these environments because they require a high computational cost and May cause delays in communications. Given the lack of security mechanisms for WSN, studies have been conducted by the academic community and research so that we all can enjoy the Internet of things safely.

Using IP technology is a great advantage, since there is already a mature infrastructure, with immense open protocols and several approaches to management and security, but these protocols are quite complex for the context of wireless sensor networks, so the IETF (Internet Engineering Task Force) studied mechanisms to use IPv6 in low-power networks developing new protocols for IoT. In this research, we will consider internet environments of things with low-power protocols 6LoWPAN (IPv6 Low-Power Wireless Personal Area Networks) defined by RFC 4944, CoAP (Constrained Application Protocol) defined by RFC 7252, and the RLP protocol (Routing Protocol for Low-Power and Loss Networks) defined by RFC 6550. These protocols help to Integrate WSN with the internet.

Because the WSN are connected to the Internet, they are vulnerable to attacks originating from the Internet, and the lack of security mechanisms for wireless sensor networks has aroused the interest of the academic and research community for the discovery of mechanisms that allow the integration of WSN with the internet in a secure manner. In many implementations, standard solutions such as the implementation of a firewall are insufficient to protect the systems, hence the need to strengthen the security of these systems with intrusion detection systems to detect passive or active attacks

The outcome of this study will culminate in the development of an anomaly-based IDS for an IoT that works at the protocol layer of applications, CoAP.

We will implement a classifier based on neural networks similar to another already existing based on SVM (Support-Vector Machines), and the performance of both will be compared since we intend to improve the solutions to the problem raised.

The metrics used to evaluate our IDS will be those used for a generic binary classification problem because the system must be able to evaluate whether or not an action is an intrusion into the system.

Keywords: CoAP, 6LoWPAN, RLP, IoT, WSN, Machine learning, IDS.

Índice

LISTA DE ACRÓNIMOS

IEEE Institute of Electrical and Electronics Engineers

DODAG Destination Oriented Directed Acyclic Graph

NIDS Network Intrusion Detection System

RFC Request for Comments

MIT Massachusetts Institute of Technology

OS Operating System

IP Internet Protocol

DTLS Datagram Transport Layer Security

IGRP Interior Gateway Routing Protocol

PLC Power Line Communications

HIDS Host Intrusion Detection System

CoAP Constrained Application Protocol

REST Representational State Transfer

RFID Radio Frequency Identification

TCP Transmission Control Protocol

HTTP Hypertext Transfer Protocol

IETF International Engineering Task Force

NFC Near Field Communication

URI Uniform Resource Identifier

IPv6 Internet Protocol Version 6

IPS Intrusion Prevention System

OSPF Open Shortest Path First

SVM Support Vector Machine

UDP User Datagram Protocol

WSN Wireless Sensor Network

KNN K-Nearest Neighbours

DAG Directed Acyclic Graph

TLS Transport Layer Security

IDS Intrusion Detection System

RPL IPv6 Routing Protocol for Low Power and Lossy Networks

6BR 6LoWPAN Border Router

RAM Random Access Memory

FIT Future Internet of Things

RSA Rivest–Shamir–Adleman

CPU Central Processing Unit

ANN Artificial Neural Network

FTP File Transfer Protocol

ROM Read-only memory

ACK Acknowledgement

IoT Internet of Things

DoS Denial of Service

QoS Quality of Service

AI Artificial Intelligence

Lista de Figuras

Lista de Tabelas

Lista de Imagens

Capítulo 1

1. Introdução

Nesta sessão apresentamos o contexto e o objetivo deste trabalho de investigação, mostramos a planificação de trabalho seguida durante o semestre bem como a estrutura deste documento.

1.2 Contexto

A IoT (Internet of Things) é um conceito atual no ramo das tecnologias de informação em que objetos para o uso quotidiano dotados de inteligência computacional são usados para automatizar tarefas difíceis ou tediosas para os humanos, por meio da sua capacidade de compartilhar, coletar, e processar informação sem intervenção humana direta [1], dentre as muitas aplicações destes dispositivos são mais populares as afetas aos carros autónomos, casas inteligentes, segurança residencial, dentre as mais diversas tarefas, estando estes conectados com a internet [1][5].

Nas redes de internet das coisas podemos encontrar vários pequenos dispositivos interligados para de forma compacta realizarem uma tarefa específica [1]. A miniaturização dos componentes eletrónicos sempre foi um objetivo a atingir desde os primórdios da computação, e hoje é uma realidade que facilita o gerenciamento destes equipamentos e a redução dos custos de aquisição dos mesmos. Mas se por um lado chegamos a um estágio em que temos dispositivos pequenos, com preços baixos devido à um vasto leque de fabricantes [11], por outro temos o poder computacional destes dispositivos reduzidos e o que impede que sejam aplicados os mecanismos de segurança tradicionais, visto que estes exigem algum poder computacional por parte dos equipamentos [17]. A falta de mecanismos de segurança para WSN levou o IETF (Internet Engineering Task Force) a desenvolver mecanismos para se usar IPv6 em redes de baixa potência desenvolvendo uma nova pilha protocolar dispositivos Low-Power [6], dentre os quais o 6LoWPAN, CoAP, RLP. Estes protocolos ajudam a interligar as WSN a internet.

Por estarem conectados com a internet as WSN ficam expostas a vulnerabilidades oriundas da internet por meio do protocolo IPv6. As soluções existentes para assegurar a rede tradicional mostram-se ineficazes no contexto da IoT (Internet of Things) devido às especificidades destes ambientes. Estudos recentes demonstram que mesmo os equipamentos supostamente seguros por criptografia tradicional podem sofrer vulnerabilidades [7] , alguns protocolos de segurança como o TLS (Transport Layer Security) e o DTLS (Datagram Transport Layer Security) são aplicados em algumas implementações, mais têm sido insuficientes para proteger as redes de IoT devido às mais diversas vulnerabilidades apresentadas em [7], daí a necessidade de se recorrer a mecanismos de proteção mais robustos como os sistemas de deteção de intrusão (IDS).

E de suma importância garantir a segurança em IoT a fim de que a arquitetura atual de comunicações e segurança da internet cresça para suportar a IoT, e protocolos como o 6LoWPAN e o CoAP permitem dispor já de uma pilha protocolar para a integração de redes de sensores com a internet utilizando mecanismos standard de comunicação (protocolos baseados no IP). [8]

1.1 Objetivo

Os objetivos gerais deste estágio cingem-se no estudo e aplicação de mecanismos de segurança em ambientes de comunicação com sensores inteligentes em conexão com a internet, utilizando protocolos IP (Internet Protocol) baseados na tecnologia 6LoWPAN (IPv6 Low-Power Wireless Personal Área Networks).

O objetivo destes mecanismos de segurança será o de contribuir para a implementação bem-sucedida de ambientes de comunicação que permitam no futuro a integração de aplicações sensoriais ubíquas com a internet, em particular ao nível da deteção de intrusões na presença de comunicações 6LoWPAN [16], RPL [18] (Routing Protocol for Low-Power and Loss Networks) definido pelo RFC 6550 e o CoAP [19] (Constrained Application Protocol) definido pelo RFC 7252 de forma integrada.

Propomos criar um classificador baseado em redes neuronais para a deteção de intrusão em CoAP, e comparar com o desempenho do classificador baseado em SVM [10].

O IDS (Intrusion Detection System) que há de ser implementado neste trabalho é semelhante ao construído em [10], pois pretendemos fazer um estudo comparativo entre os sistemas, os comportamentos anómalos do sistema foram implementados em [10], no sistema operacional *contiki* CoAP [34].

A arquitetura desenvolvida baseia-se na ideia de que um único nó, o que há de ser responsável pelo módulo IDS, é capaz de executar um *sniffer* em conjunto com um interpretador de linguagem de alto nível.

Estudaremos as principais soluções de segurança atualmente utilizadas e em desenvolvimento para a IoT, dando sequência ao trabalho de investigação desenvolvido em [10], desenvolvendo um IDS baseado em anomalias, para IoT operando na camada de aplicação, CoAP e da tecnologia 6LoWPAN, e de modos a cumprirmos cabalmente com os objetivos gerais desta pesquisa, realizaremos as tarefas abaixo descritas:

- Treinar um novo classificador para detetar instruções baseado em redes neuronais
- Detetar ataques do tipo DoS nos dados oriundos da rede de sensores
- Comparar os resultados obtidos nesta implementação com os da implementação passada.

1.3 Planificação.

A planificação das atividades realizadas durante as duas etapas deste estágio pode ser consultada por meio dos diagramas abaixo:

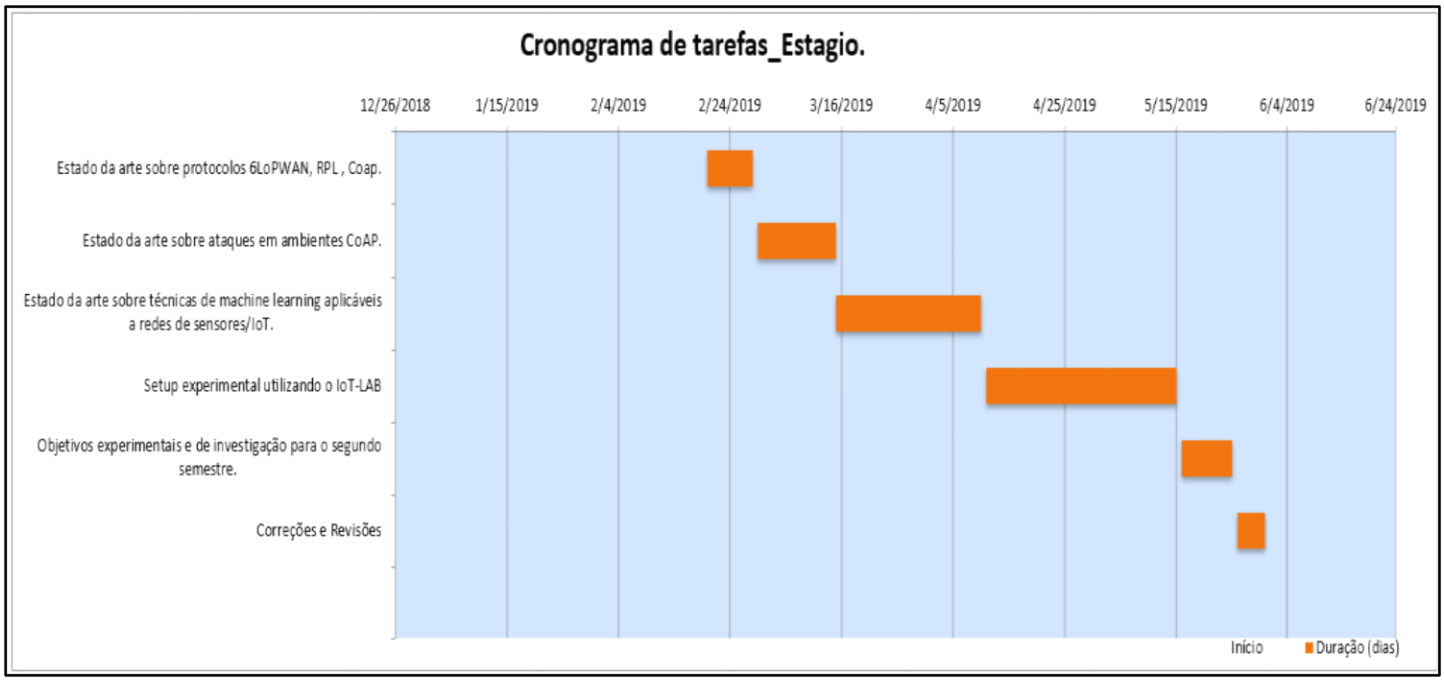


Figura 1:Cronograma de atividades do primeiro semestre

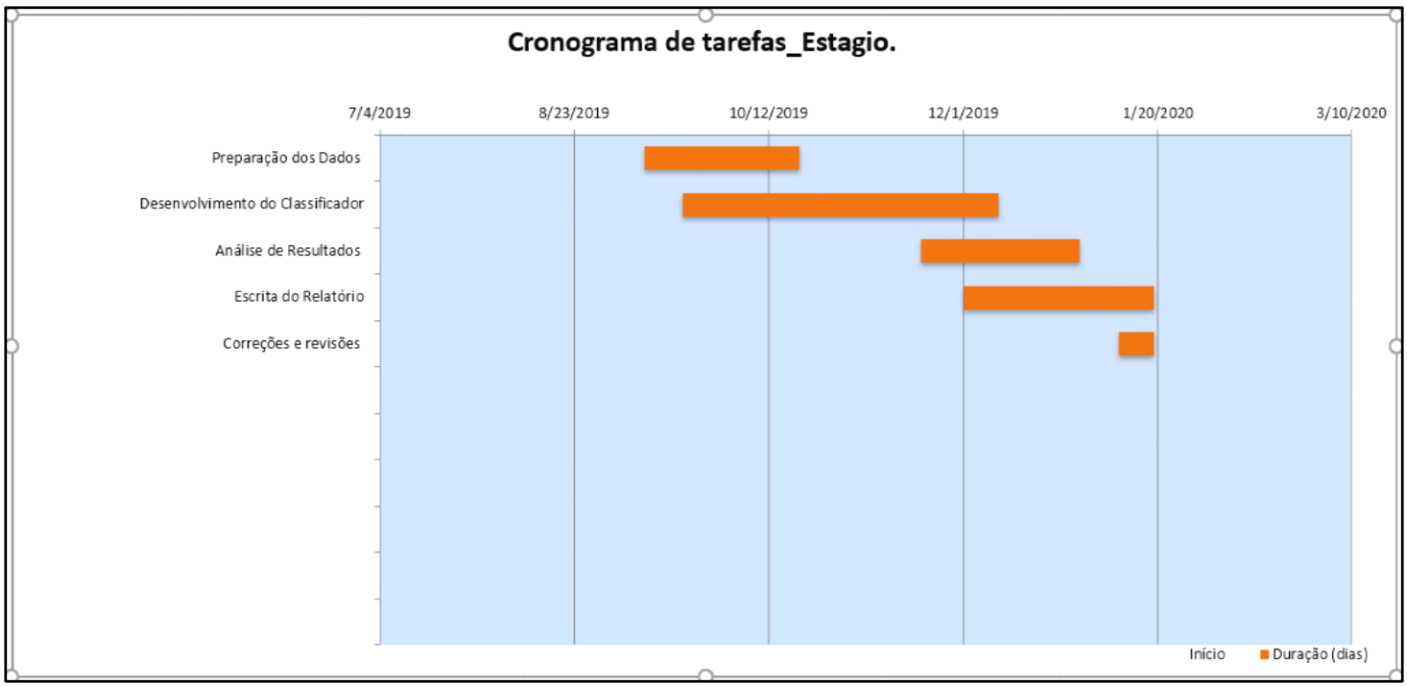


Figura 2:Cronograma das atividades do segundo semestre

1.4 Estrutura do Documento

Capítulo 1 - Introdução

Nesta secção apresentamos ao leitor o assunto em estudo nesta pesquisa bem como o seu contexto, definimos o âmbito da pesquisa por meio do objetivo, e a planificação do trabalho realizado.

Capítulo 2 - Fundamentação teórica

Nesta seção apresentamos os conceitos fundamentais para que o leitor consiga perceber de forma mais abrangente o trabalho que está a ser desenvolvido.

Capítulo 3 - Estado da arte

Nesta seção apresentamos o estado da arte sobre os protocolos de IoT, nomeadamente o protocolo RLP, 6LoWPAN, e CoAP, enunciamos as técnicas de aprendizagem de máquina mais usadas em IoT, especificamos quais destas são usadas no contexto específico de deteção de intrusão e qual será implementada. Por fim apresentamos o estado da arte sobre ataques em ambientes CoAP.

Capítulo 4 – Implementação e estratégia de Avaliação

Neste capítulo descrevemos o processo de implementação da solução proposta

Capítulo 5 – Análise dos Resultados

Nesta seção apresentamos os resultados alcançados durante os testes e avaliamos o impacto dos mesmos

Capítulo 6 - Conclusões e Trabalhos futuros

Este último capítulo apresenta de forma resumida as conclusões tiradas desta pesquisa e apresenta sugestões para estudos futuros

Capítulo 2

Fundamentação Teórica

2.1 Conceitos Fundamentais

Podemos encontrar os conceitos aqui apresentados em:

Intrusão: qualquer tentativa de comprometer a política de segurança de uma rede ou sistema, (integridade, confidencialidade, privacidade, autenticidade ou não-repúdio) [9].

Deteção de intrusão: a monitorização dos eventos que acontecem em um sistema de computador ou uma rede e posterior análise deles para sinais de intrusões [10]**.**

Firewall: refere-se a um sistema de segurança de rede que monitoriza o tráfego vindo e saindo da rede. É o *firewall* que decide se permite ou intercepta tráfego específico de acordo com um conjunto de regras predefinidas. Os *firewalls* constituem, então, um muro entre redes internas que podem ser confiáveis e não confiáveis fora das redes, estes podem ser um elemento de hardware, software ou ambos [9].

Alerta: uma indicação de que um sistema está sob ataque [9].

Sistema de Deteção de Intrusão (IDS): o *hardware* ou o *software* responsável por automatizar a deteção de intrusões [9].

Sistema de prevenção de Intrusão: um sistema que herda todas as capacidades do IDS e tem a capacidade de bloquear as atividades maliciosas [9].

Vulnerabilidade: qualquer condição ou estado que representa uma fraqueza de um elemento do sistema [7].

Ataque: qualquer ação interna ou externa ao sistema que visa explorar as vulnerabilidades do sistema e consequentemente quebrar os pilares da segurança do mesmo [9].

Aprendizagem de máquina: O ramo da inteligência artificial que se dedica a estudar técnicas para que os sistemas sejam capazes de aprender sem intervenção humana e direita [2].

(TP) Verdadeiro Positivo: Quando o IDS deteta corretamente uma tentativa de invasão ao sistema [2].

(TN) Verdadeiro Negativo: Quando o IDS não deteta corretamente atividades que não representam uma tentativa de invasão ao sistema [2].

(FP) Falso Positivo: Quando o IDS deteta incorretamente uma intrusão não existente [2].

(FN) Falso Negativo: Quando o IDS não consegue detetar uma intrusão ao sistema [2].

Confidencialidade: refere-se à capacidade de proteger informações estáticas ou em trânsito na rede contra acesso não autorizados [4].
Integridade: refere-se à capacidade de garantir que as informações que trafegam na rede não foram alteradas

[4].

Autenticidade: capacidade de validar a identidade de uma entidade [4].

Não-repúdio: garantir que uma entidade não possa negar a responsabilidade de uma determinada ação [4].

Controle de acesso: refere-se à capacidade do sistema de restringir acesso apenas a entidade não autorizadas [4].

2.2 O Ambiente de experimentação (IoT- LAB)

O *IoT-LAB* [36] oferece uma infraestrutura de grande escala adequada para testar pequenos dispositivos de sensores sem fio e objetos de comunicação heterogêneos.

O ambiente de experimentação (*IoT-LAB*) oferece controle dos nós da rede e acesso direto aos *gateways* conectados o que permite aos seus utilizadores monitorizar o consumo energético dos nós bem como outras métricas relacionadas ao sistema, tais como: atraso fim a fim, taxa de transferência ou sobrecarga. O *IoT-LAB* oferece um ambiente de fácil implementação, avaliação, coleta e análise de resultados [36].

O *IoT-LAB* faz parte da plataforma FIT (Future Internet of the Things) que é um conjunto de componentes complementares que permitem a experimentação de serviços inovadores para utilizadores na academia ou na indústria, oferecendo um meio para experimentação de comunicações sem fios para a rede e em camadas de aplicativos, o que acelera o *design* de redes avançadas para o Internet do futuro", e foi a plataforma escolhida para se obter o conjunto de dados usado para os nossos testes [36].

2.3 O sistema operativo Contiki

O Contiki [34] é um sistema operativo de código aberto para a internet das coisas desenvolvido por Adam Dunkels em 2003, em linguagem C. O *Contiki* foi desenvolvido para conectar pequenos microcontroladores com capacidade de processamento, memória e largura de banda restritos, suporta IPv4 (Internet Protocol Version 4), IPv6 (Internet Protocol Version 6) e 6LoWPAN.

O *Contiki* possui um simulador de rede chamado CooJa desenhado para WSN, e não obstante ser multitarefa e suportar a pilha protocolar TCP/IP (Transmission Control Protocol/Internet Protocol) o *Contiki* só precisa de alguns *Kbytes* de código e algumas centenas de bytes de RAM (Random Access Memory), O modelo de programação do *contiki* é baseado em *protothreads*, que possui características de programação orientada a eventos com o intuito de melhorar a eficiência de gestão de memória. O *contiki* tem grande aceitação por parte da comunidade académica e de investigação [34].

2.4 A Internet das Coisas

Como já mencionamos anteriormente a IoT, termo sugerido por Kevin Ashton pesquisador do MIT (Massachusetts Institute of Technology), é um conceito emergente no campo da ciência e tecnologia, que

visa interligar dispositivos eletrônicos utilizados no nosso dia a dia como sensores e atuadores a internet. O que se pretende de facto é criar um mundo onde a internet emerge em todos os lugares e coisas [11] [1].

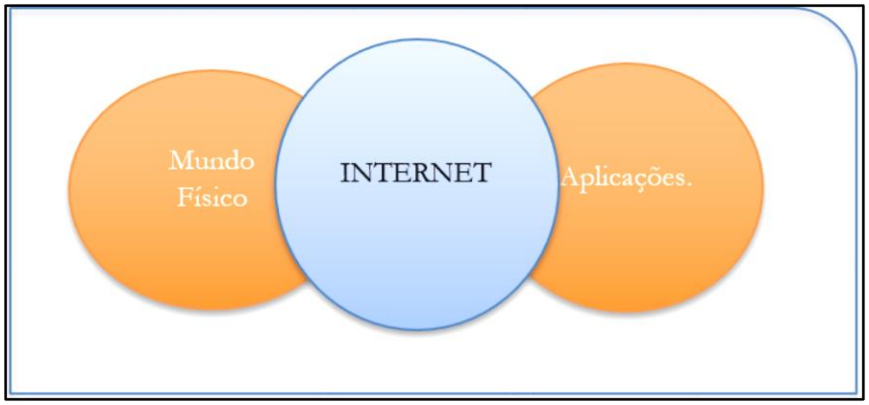


Figura 3:Visão geral da IoT

Segundo o instituto *Gartner* teremos mais de 20 bilhões de dispositivos conectados a internet até 2020 [12]. A internet convencional utiliza o modelo de referência TCP/IP, a internet das coisas segue o mesmo modelo usando o protocolo TCP/IP como referência, utilizando 4 camadas, que são: rede, Internet, transporte e aplicação. Devido às limitações das WSN, as soluções a nível de protocolos para a internet convencional mostram-se complexas demais, pelo que urge a necessidade de se desenvolver uma nova pilha protocolar adaptada às restrições entre os quais os protocolos: LaRaWAN (Protocolo de comunicação sem fio destinado a resolver problemas de gestão energética em dispositivos da IoT), Sigfox (Tecnologia de baixo custo e de baixa potência usada para conectar sensores da IoT), NFC (Near Field Communication), esta tecnologia permite comunicação entre dispositivos sem fio de forma segura, 6LoWPAN, Zigbee (protocolo para comunicação sem fio de baixo potência) ,Bluetooth (Tecnologia que permite comunicação entre dispositivos sem fio com baixo consumo de energia e alcance entre equipamentos), CoAP, RLP [13] .

O protocolo DTLS (Datagram Transport Layer Security) foi criado para lidar com as dificuldades de confiança do UDP (User Datagram Protocol) e tem uma particular atenção na literatura por ser leve e permitir fluxo entre os dados [8], sendo usado para a segurança em muitas aplicações IoT, mas é insuficiente para proteger as comunicações pois é vulnerável a ataques DoS (Denial of Service), roubo de chaves privadas RSA (Rivest–Shamir–Adleman), problemas de usabilidade e muitos outros descritos em [7], na tabela 1 podemos observar categorização dos protocolos em camadas seguindo a visão protocolar TCP/IP.

Modelo de referência TCP/IP	Protocolos
4-Aplicação	CoAP
3- Transporte	UDP, DTLS
2- Rede	6LoWPAN, RLP
1-Física/ Enlace	*RFID, IEEE 802.15.4*

Tabela 1:Categorização dos protocolos em camadas[4]

O componente principal de uma rede de sensores sem fios é o nó sensor sem fio, e ao conjunto de vários nós sensores sem fio denominamos de rede de sensores sem fio. Um nó sensor é basicamente um pequeno computador de baixo custo, com restrições de processamento, memória e armazenamento de energia, e o conceito de IoT conecta estas redes de forma transparente com a internet, existem vários tipos de sensores disponíveis comercialmente, desde sensores de temperatura, ruído, luminosidade, gás, presença e outros [11].

As redes de sensores são usadas nas mais diversas aplicações, tais como: saúde, controle e segurança rodoviária, aplicações militares, ambiente, segurança pública e outras, pelo que é de suma importância garantir a segurança das WSN [1][5].

Não obstante, exista uma grande diversidade de sensores, estes podem ser divididos em duas categorias principais, de acordo com os seus recursos [17]: Os nós sensores de classe 1 e de classe 2, como podemos observar na tabela abaixo. A implementação que gerou o *dataSet* que será usado nesta pesquisa considerou nós sensores de classe 1, nomeadamente o TeloB [17] e MicaZ[17][10].

Classe	**Memória Ram**	**Memória Flash**
Classe 1	< 10 kB	< 100 kB
Classe 2	< 50 KB	< 250 kB

Tabela 2:Categorização dos nós sensores[17]

2.5 Sistemas de Deteção de Intrusão

Um *firewall* constitui a primeira linha de proteção dos sistemas e redes informáticas [4], estes possuem a desvantagem de serem simplesmente dotados de mecanismos para a análise de intrusões aos sistemas baseados em protocolos e portos de comunicação. Sendo que as modalidades de ataques a sistemas têm evoluído, é imperioso que os sistemas se equipam com mecanismos mais robustos de segurança, pelo que é necessário a implementação de uma segunda linha de proteção, os IDS, visto que estes podem fazer a avaliação não apenas por meio de protocolos e portos, mas também pelo conteúdo da informação e outros critérios dependendo do tipo de IDS a ser implementado [14].

Para detetar essas ameaças maliciosas, é essencial adotar um sistema de deteção de intrusões (IDS). Um IDS pode ser usado junto com um sistema de prevenção de intrusão (IPS) e/ou um *firewall* para proteger os sistemas de computador contra-ataques que violam as políticas de segurança da rede, como integridade, confidencialidade, privacidade, autenticidade, não-repúdio e controle de acesso.

Assim, um IDS constitui basicamente um sistema automático que pode monitorizar uma rede ou sistema, de forma a detetar ações suspeitas destinadas a violar as regras de segurança definidas pelo administrador. Uma vez que uma atividade maliciosa é detetada, esta pode ser reportada ao administrador e consequentemente bloqueada. Os sistemas IDS podem usar uma técnica específica ou até mesmo combinar várias técnicas para detetar ataques em conexões suspeitas (conhecidas ou não). Ameaças conhecidas também chamadas de assinaturas de ataque podem ser facilmente bloqueadas combinando a adição de um *firewall* ao nosso IDS através de regras predefinidas. Os ataques desconhecidos dependem da capacidade e especificidades do IDS usado para proteger o sistema. A figura 4 mostra de forma simplificada a arquitetura de um sistema de deteção de intrusão.

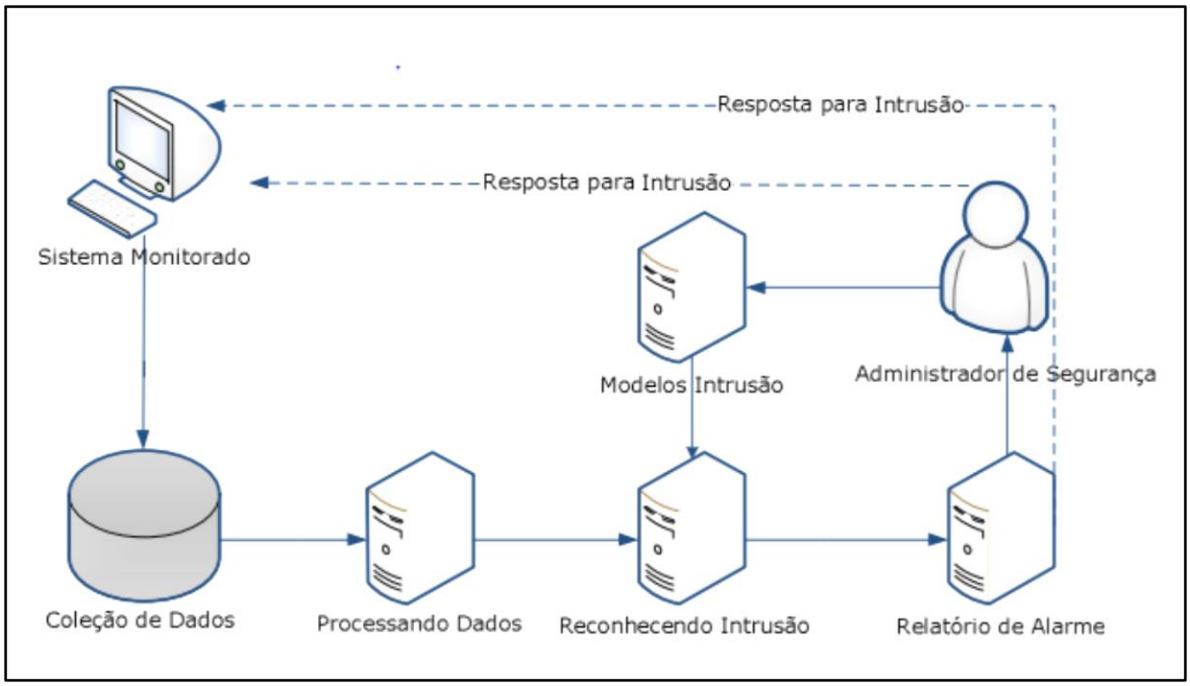


Figura 4:Arquitetura de um sistema de deteção de intrusão [2]

A princípio os dados são coletados do sistema a monitorizar, e depois os mesmos são processados para a devida avaliação. A deteção de ataques desconhecidos não se faz de modo tão trivial como a dos ataques conhecidos, sendo para isto necessário a aplicação de diferentes modelos de ataques a aplicar de acordo com as características do sistema.

2.5.1 Modos de Deteção de Intrusão

Abaixo descrevemos três abordagens usados pelos IDS, para a deteção de intrusão:

- **Deteção Por Assinatura**

Detetores baseados em assinatura detetam ataques conhecidos, ou seja, buscam por padrões conhecidos de ataques também chamadas de assinaturas de ataque. Uma assinatura corresponde a um padrão definido de ataque, um exemplo fácil para criar um padrão de ataque é a tentativa de um acesso remoto de um usuário que digitaliza mais de 3 vezes a palavra passe errada, pelas assinaturas registradas nos ficheiros de *logs*, um alarme ao administrador é acionado e naturalmente o acesso a este usuário será impedido [4]. Uma das grandes vantagens deste tipo de deteção é a sua eficácia na deteção dos ataques, visto que já estão predefinidos os modelos ou formas comportamentais dos atacantes, gerando assim um número baixo de falsos positivos. A desvantagem de detetores baseados em assinaturas consiste no facto de serem incapazes de detetar ataques desconhecidos ou variantes de um mesmo ataque, [14].

- **Deteção por Anomalias**

Diferente da deteção por assinatura que se destina a avaliar ou bloquear ataques conhecidos, os IDS que usam deteção baseados em assinaturas focam-se em definir aquilo que é o padrão normal de funcionamento da rede ou dos usuários, definindo assim um perfil de comportamento, este padrão é definido por meio de coleta de dados durante um período de teste [14]. Depois de definido o padrão, todo o comportamento anômalo nos nós da rede, conexões, ou até mesmo em atividades dos usuários, será considerada um possível ataque à rede ou ao sistema a proteger, pois este tipo de deteção baseia-se no conceito de que ataques apresentam características diferentes do comportamento normal do sistema [15].

Esta abordagem tem a vantagem de ser capaz de detetar ataques não conhecidos, serve como base para recolha de informação para que se criem padrões que podem ser utilizados como assinaturas de ataques, sendo a grande desvantagem produção de um número elevado de alarmes falsos, que se deve a comportamentos inesperados dos sistemas e das atividades dos usuários [14] [15].

- **Ataques baseada em especificação**

Esta abordagem apresenta maior complexidade do que as anteriores, visto que a análise é feita nas camadas abaixo da camada da aplicação da pilha protocolar TCP/IP, ou até mesmo a nível do controle do sistema operativo [14].

Esta técnica cinge-se apenas no manuseio correto de um protocolo ou programa monitorando o funcionamento do programa de acordo as definições pré-estabelecidas.

Esta abordagem deteta ataques previamente conhecidos, e destina-se a uma única aplicação, sendo contanto de aplicação restrita e não é amplamente divulgado devido a sua complexidade.

Tem como vantagem o facto de apresentar um número muito baixo de falsos positivos, pois requer um conhecimento amplo e profundo das atividades realizadas pela aplicação ou programa, da topologia da rede, bem como do sistema operativo, a título de exemplo podemos pensar em uma análise de ligações de um servidor FTP (File Transfer Protocol) ao núcleo do sistema operativo, com o intuito de encontrar possíveis alterações com fins maliciosos [14].

2.5.2 Tipos de IDS

Os IDS podem usar diferentes abordagens para análise do tráfego, e existem basicamente dois tipos de sistemas de deteção de intrusão, os baseados em rede e os baseados em *host*, descritos a seguir:

- **IDS baseados em rede**

Os sistemas *IDS* de rede, *NIDS* (Network Intrusion Detection System), analisam o tráfego numa ou em várias interfaces de rede com o intuito de detetar os ataques previamente conhecidos definidos por meio das assinaturas de ataques.

Para a captura do tráfego, usa-se uma interface no modo promíscuo, onde a placa de rede deverá ler todos os pacotes detetados na rede [14].

Os sistemas NIDS têm vantagens do ponto de vista econômico. Isso é verdade porque os NIDS podem ser colocados em zonas estratégicas da rede, protegendo vários host e recursos ao mesmo tempo. Os sistemas NIDS também possuem o potencial de analisar os campos de dados dos pacotes IP, o que permite a deteção de ataques de negação de serviço (DoS) [14]. E têm a desvantagem de gerar um número alto de falsos positivos.

- **IDS baseados em host**

Os sistemas de deteção baseados em host HIDS (Host Intrusion Detection System), avaliam o estado de um ou mais servidores, com a finalidade de encontrar comportamentos suspeitos dentro do sistema operativo. Sistemas de deteção de intrusão por host são dotados de várias estratégias de modos a detetar, aplicações com atividades anormais, informações relevantes nos ficheiros de *log* do sistema, ou o corolário das avaliações a nível de integridade, em ficheiros e programas críticos do sistema, estas verificações levam em consideração a análise de rede e a entrada ou a saída do host por meio das suas interfaces [14].

Tal como nos IDS de rede, é necessário que a interface de rede esteja configurada no modo promíscuo, a fim de que a placa de rede seja capaz de ler todos os pacotes detetados na rede, como condição indispensável para a deteção de intrusões. Alguns sistemas são dotados da habilidade de fazer a análise em volta da pilha protocolar TCP/IP, permitindo que os ataques sejam barrados, antes de alcançarem as aplicações alvos, ou ao sistema operativo [14[15].

HIDS têm a vantagem de não ter acesso a cargas de pacotes IP e em um ponto de vista de análise *forense*, se um host for comprometido, existe a possibilidade de recuperar os arquivos de *log* NIDS e oferecem a possibilidade de visualizar os ataques, especificamente em redes onde *switches* ou protocolos de comunicação criptografados dificultam sua implementação [14].

Capítulo 3

Estado da Arte

Nesta seção apresentamos o estado da arte sobre os protocolos 6LoWPAN, RPL, CoAP, sobre ataques em ambientes CoAP e a respeito das técnicas de aprendizagem de máquina aplicadas a internet das coisas a compreensão destes tópicos é importante para que o caro leitor entenda a natureza dos dados que serão usados nos testes, e o algoritmo de aprendizagem de máquina selecionado para resolver a problemática da nossa pesquisa.

3.1 Estado da arte sobre protocolos 6LoWPAN, RPL, CoAP

3.1.1 6LoWPAN

Segundo o RFC 4944 o 6LoWPAN [16] é um padrão aberto definido pelo IETF, no padrão RFC 4944. O protocolo 6LoWPAN fornece um meio de transportar dados de pacotes na forma de IPv6 sobre IEEE 802.15.4 [39], e outras redes. O sistema 6LoWPAN é usado para uma variedade de aplicações, incluindo redes de sensores sem fios. essa forma de rede de sensores sem fios envia dados como pacotes e usa IPv6, fornecendo a base para o nome IPv6 em redes de área pessoal sem fio de baixa potência.

Não obstante, o 6LoWPAN tenha sido originalmente projetado para se basear no IEEE 802.15.4, é um padrão que define as camadas inferiores para um sistema sem fio de baixa potência de 2,4 GHz, ele está sendo desenvolvido e adaptado para trabalhar com muitos outros portadores sem fio, incluindo *Bluetooth*, controle de linha de energia, PLC (PowerLine Comunications) e *Wi-Fi* de baixa potência.

De realçar que o 6LowPAN tem grande relevância no contexto das redes de sensores sem fio, visto que por meio deste é possível garantir comunicação entre dispositivos da uma rede IP e uma rede de sensores de forma transparente e homogénea [17].

3.1.2 - RPL

Como sabemos, nos cenários de IoT encontramos conectados vários pequenos dispositivos, estes possuem restrições a nível de armazenamento e capacidade de processamento, e os protocolos de encaminhamento tradicionais da internet como o RIP (Routing Information Protocol), IGRP (Interior Gateway Routing Protocol), OSPF (Open Shortest Path First), e outros deixam de ser eficazes para as necessidades e restrições do ambiente da internet das coisas, daí o esforço da *IETF* em desenvolver novos protocolos que se adequem a IoT, e entre esses protocolos está o RPL (IPv6 Routing Protocol for Low Power and Lossy Networks) [18] definido pelo RFC 6550.

Segundo o RFC 6550 o RPL é um protocolo de encaminhamento que diferente dos protocolos tradicionais da internet baseados em tabelas, utiliza a abordagem de vetor de distâncias para a definição das rotas, construindo uma árvore de encaminhamento, com base no conceito de DAG (Directed Acyclic Graph) desenvolvido para ambientes com restrições a nível de recursos computacionais, e com grandes variações de largura de banda, este protocolo considera o período em que os nós não são acessados, oferecendo rotas alternativas além das padronizadas a fim de que o tráfego de informação na rede não seja comprometido.

A diferença entre o RPL e os demais protocolos baseados em topologias de árvores de encaminhamento, cinge-se no facto de que: a abordagem baseada em DAG permite que um nó da rede se associe a mais de um nó, criando assim um ou mais destinos orientados, resultando num DODAG (Destination-Oriented Directed Acyclic Graph). A associação de um ou mais DODAG, forma uma instância RPL que recebe um identificador (RPL instance ID), sendo que um nó da rede pode ser parte de várias instâncias RPL, mas apenas de um DODAG.

O protocolo define métricas como a contagem do número de saltos, energia, para definir como o nó vai afetar o encaminhamento e uma função objetiva para otimizar o encaminhamento.

3.1.3 O CoAP

Com o advento da IoT, onde temos vários dispositivos a comunicar na rede em um ambiente bastante restrito, o tradicional e bem sucedido HTTP **(**Hypertext Transfer Protocol) definido pelo RFC 2616 [53] que é o padrão para *web*, mostrou-se complexo demais e ineficiente para o ambiente restrito da IoT, o que motivou o CORE (Constrained *RESTful Environments)*, grupo de trabalho da IETF a desenvolver um protocolo mais leve que se adapta-se às restrições afetas a internet das coisas, O CoAP.

O CoAP, definido pelo RFC 7252 [19] opera na camada de aplicação, usa o protocolo UDP (User Datagram Protocol) e serve para a transferência de informações, foi desenhado para ser usado na *Web* em ambientes *LowPower* , é muito semelhante ao HTTP e pode funcionar na posição de cliente ou de servidor numa ligação M2M (Machine-to-Machine)

Segundo o RFC 7252 [19] Este protocolo funciona com o modelo de cliente/servidor entre *endpoints* de aplicativos, suporta os conceitos fundamentais da *Web,* tais como: URI (Uniform Resource Identifier) e tipos de *mídia* da internet. O CoAP foi concebido a fim de interagir amigavelmente com HTTP para integração com a *Web*. Numa solicitação CoAP tal como em HTTP, um cliente solicita uma ação ao servidor usando métodos definidos pelo protocolo em um recurso (identificado por um URI) a um servidor, este por sua vez envia uma resposta ao cliente segundo regras pré-definidas pelo protocolo. O CoAP não foi desenvolvido para compactar cegamente o HTTP, mas sim para realizar um subconjunto de REST (Representational State Transfer), comum com HTTP, otimizado para aplicativos M2M**,** embora também forneça recursos para M2M, como suporte a *multicasting* (referente ao envio de informação para vários destinos ao mesmo tempo) , descoberta integrada de serviços, assincronismo nas trocas de mensagens, sobrecarga muito baixa, é dotado de simplicidade para ambientes restritos.

Como já referimos acima o CoAP é extremamente semelhante ao HTTP, mas usa uma camada de mensagens abstrata com suporte opcional de confiabilidade, sendo que o CoAP especifica um conjunto mínimo de instruções *REST* incluindo *POST (* Cria um novo recurso) *, GET* (Obtém informações de um recurso identificado *), PUT* (Atualiza o recurso) *e DELETE* (Exclui o recurso)*,* uma outra diferença é o facto de que a comunicação com o protocolo CoAP acontece de forma assíncrona, ou seja a conexão não é efetivada antes da troca de mensagens, e por último este usa 4 tipos de mensagens, sendo que as duas primeiras mensagens funcionam como indicadores de QoS(Quality of Service).

- CON(Confirmável)
- NON (Não confirmável)
- ACK (Confirmação)
- RESET.

Abaixo a ilustração da transmissão de uma mensagem confiável e outra não confiável do protocolo CoAP.

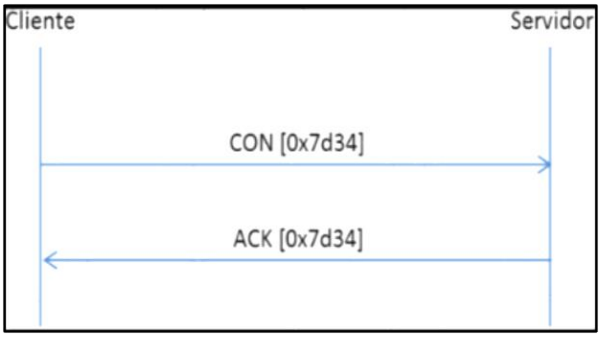


Figura 5:Transmissão de uma mensagem confiável[17]

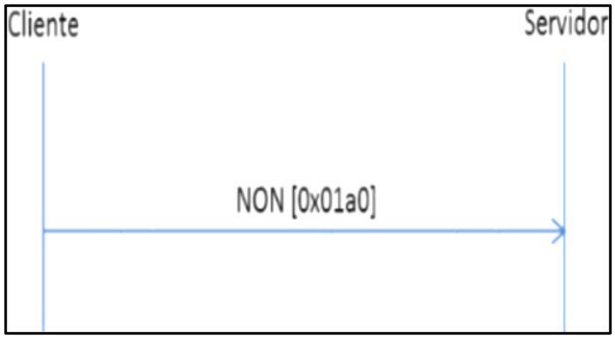


Figura 6:Transmissão de uma mensagem não confiável[17]

Podemos perceber com mais detalhes diferentes aspetos entre os protocolos HTTP e CoAP analisando a figura 2, bem como a camada de abstração do CoAP pensado nele numa abordagem de duas camadas, uma das quais é a camada que lida com o UDP chamada de camada de mensagens e a de camada de solicitações resposta, como nos mostra a figura 7.

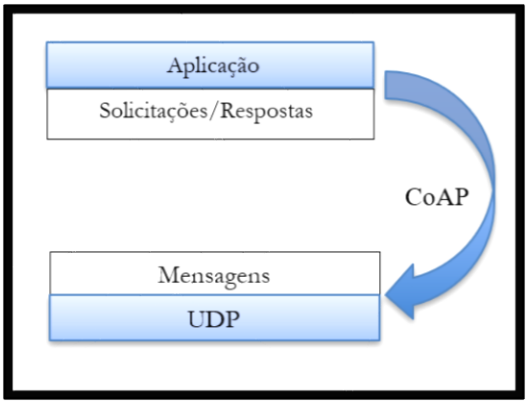


Figura 7:Camada de abstração do CoAP[17]

3.2 Estado da arte sobre ataques em ambientes CoAP

O Ataque de negação de serviço é um evento que ocorre quando um atacante inibe usuários legítimos de aceder a recursos do servidor, sobrecarregando o servidor com um número de solicitações acima daquelas que este é capaz de suportar, ou enviando informações específicas que podem gerar falhas no sistema. DoS são detetados com mais facilidade por IDS baseados em anomalias, a abordagem que será implementada neste trabalho.

3.2.1 Deteção de Ataques de Negação de Serviço

Na literatura podemos encontrar vários tipos de ataques DoS [3]

Ataque de interferência: Este tipo de ataque explora a transmissão de interferência nas frequências de sinais de rádio usada pela rede de sensores.

Ataques de clonagem de sensores ou atuadores: Este tipo de ataque consiste na troca de nós da rede durante o período a manutenção do sistema como por exemplo numa atualização de *software* no mesmo nó físico, sendo também possível um nó ser capturado enquanto está em operação. Nestas situações é possível que se manifestem novas falhas de segurança e extrações de alguns parâmetros de segurança.

Ataque de Encaminhamento: Por meio de retransmissão, falsificação ou alteração de informações de encaminhamento de um conjunto de nós, é possível criar *loops* de comunicação.

Ataques de aplicativos: Visto que o CoAP é um protocolo recente, é normal que haja alguma instabilidade a nível de segurança, algumas vulnerabilidades conhecidas são as inundações de SYN que é basicamente uma modalidade de ataque de negação de serviço em que o servidor recebe um número de requisições SYN maior duque as que pode suportar, *proxy* (intermediário entre o usuário e seu servidor), protocolo cruzado ou análise de protocolo.

Ataques de escutas: Este tipo de ataques pode acontecer caso haja algum descuido na comunicação por um dos elementos, como por exemplo a troca de chaves sem criptografia, e mesmo em cenários que haja criptografia podem ainda surgir análises criptográficas ou mesmo ataques de força bruta (ataque destinado a quebrar a segurança de um sistema usando toda a combinação possível entre caracteres para se descobrir a senha correta) que podem reduzir a entropia das chaves (nível de dificuldade de se quebrar uma chave) e facilitar a decodificação de informações, podendo estes evoluir para um ataque *Man-in-the-Middle* (ocorre quando uma entidade não legítima intercepta uma determinada comunicação).

Ataques de IP Spoofing: Por não existir um meio através do qual duas maquinas garantem que se reconhecem e estão dispostas a comunicarem (*Handshake*) em UDP, o nó que acede à rede pode fazer *spoofing*, ou seja um elemento da rede pode se fazer passar por outro mascarando o seu endereço de IP para enviar mensagens do tipo ACK(*Acknowledge*) [7].

Ainda em relação ao CoAP, baseada no RFC 7252 descrevemos abaixo os ataques principais a segurança do protocolo CoAP.

Proxying e Caching : Um *proxy* em uma rede é elemento *Man-in-the-Middle*, e por meio deste podem ser amplificadas algumas ameaças.

Risco de Amplificação: As respostas do protocolo CoAP são maiores em relação às requisições, facilitando assim ataques por amplificação.

Ataques Cross Protocol: Ligados à possibilidade de se usar o CoAP para atacar outros protocolos, a título de exemplo com o intuito de se passar pelo *firewall*.

Ataques de IP Spoofing: Como não há *handshake* em UDP, o nó que acede à rede pode fazer *spoofing* para enviar mensagens do tipo ACK no lugar de CON (*Confirmable)* [7].

Nós com Restrições: Quer sejam restrições a nível de memória, processamento ou energéticas, tornam difícil o uso de entropia por parte dos dispositivos, Não obstante muitos processos que precisam de entropia o façam externamente [19].

Um ataque **DDos** (Distributed Denial of Service) é ligeiramente diferente de um ataque DoS. Num DDos temos um maior número de atacantes com uma menor frequência de envio, o que dificulta a deteção deste tipo de ataque.

3.2.2 Soluções presentes na Literatura para deteção de ataques de negação de serviço

Algumas soluções estão descritas na literatura para deteção de ataques DoS a primeira descrita em [20] não é específica para o CoAP, embora seja baseada em 6LoWPAN, nesta abordagem os ataques são prevenidos ataques originados na internet, implementando o sistema de deteção, em dispositivos mais potentes, os encaminhadores de borda. A Segunda solução proposta em [21] baseia-se na deteção em redes *ebbits* [21], onde é proposto um a arquitetura de um sistema automatizado capaz de detetar intrusões em equipamentos IoT que estejam conectadas às redes *ebbits* [21]. Uma rede IoT oferece uma plataforma para equipamentos IoT remotos, fornecendo uma plataforma de segurança. A figura abaixo mostra a arquitetura da rede *ebbits*.

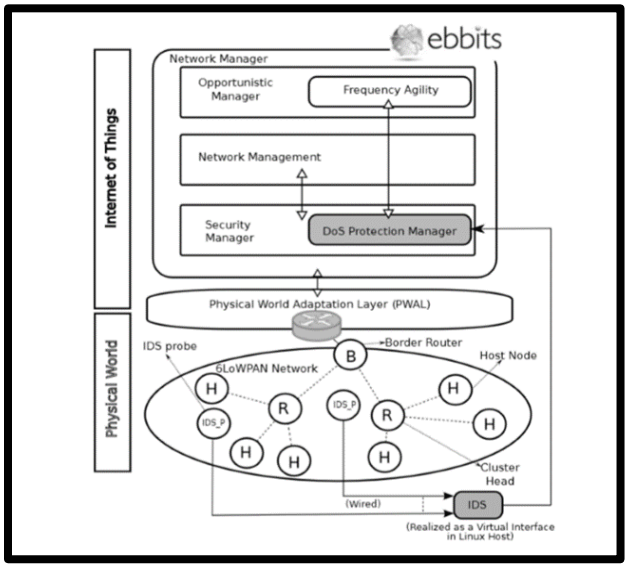


Figura 8: Gerenciador ebbits [21]

O gestor de rede contém um gestor de segurança, é responsável pela aplicação de todas as políticas de segurança, criptografia e mecanismos de segurança bem como garante segurança na comunicação entre o *ebbits manager* e o mundo físico.

Os gestores de segurança contêm dois elementos, o gerenciador de proteção DoS e o IDS, este IDS é composto por sondas com as interfaces de rede configuradas no modo promíscuo para capturar todo tráfego sem fio.

As sondas (*probes*) conectam-se a uma máquina com um IDS de código aberto ativado, o *sucurita* [53]. O gestor de proteção receberá as alertas do IDS sempre que surgir a tentativa de alguma invasão. E também vai coletar parâmetros de recursos, tais como as taxas de interferência e as perdas de pacotes dos outros gestores de *ebbits* executando dados cruzados para a confirmação de um ataque real.

A eficiência deste IDS foi avaliada por meio de testes de penetração, usando recursos de simulação do *contiki*, o sistema IDS é batizado com o nome de *Sucurita* é um sistema de deteção baseado em assinaturas onde a deteção de ataques dependerá da definição de regras específicas inseridas de forma manual pelo gestor do sistema, o que configura uma grande desvantagem para ambientes tão imprevisíveis como os de IoT [21].

Uma outra abordagem é proposta em [10], onde usa-se aprendizagem de máquina para detetar intrusão em CoAP por meio de um classificador baseado em máquinas de vetores de suporte.

Em ambientes CoAP quando um servidor 6BR (6LoWPAN Border Router) é o elemento que dá acesso a internet, e um ou mais servidores CoAP com recursos limitados são configurados, os clientes são vinculados aos servidores e têm permissão para atuar, solicitar e até mesmo observar recursos do servidor.

O IDS implementado tem a responsabilidade de detetar um nó afetado e bloquear a comunicação dos demais nós com o nó afetado, enviando mensagens de alerta para os demais nós, a fig. abaixo mostra a arquitetura implementada.

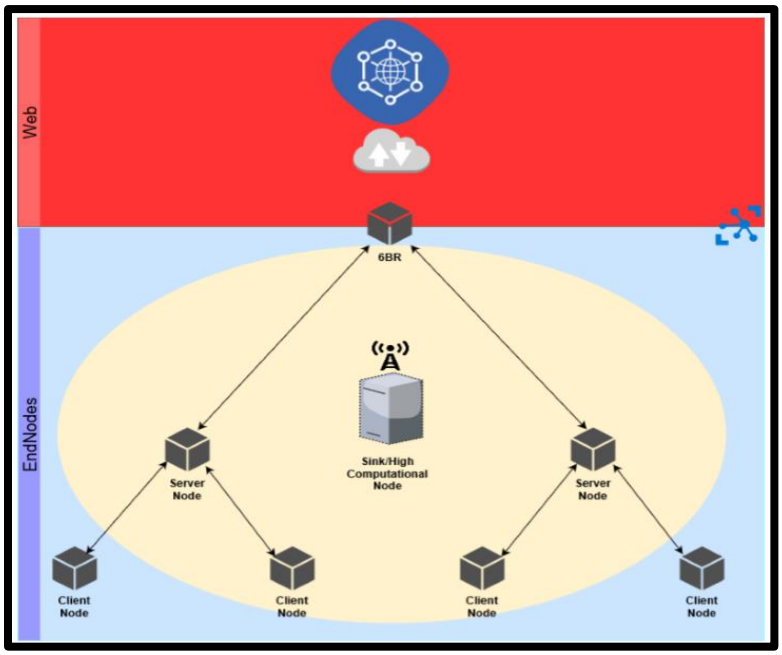


Figura 9:Arquitetura da solução proposta[10]

Os nós afetados foram identificados pela presença de comportamentos anómalos, como por exemplo um nó que perde a bateria de forma anormal, ou nós que processem mensagens indevidas, o que compromete o desempenho da rede. As seguintes anomalias foram consideradas na implementação.

- *DOS FREQ* : Nó com uma taxa maior de solicitações de CoAP no respetivo servidor.
- *DOS ACK :* Envio de mensagens de confirmação, sem nenhuma solicitação enviada.
- *WRONG URI* : Solicitação de recursos não suportáveis por um nó.
- *WRONG ACCEPT* : Envio de solicitações como opção *ACCEPT* enviada para o servidor.

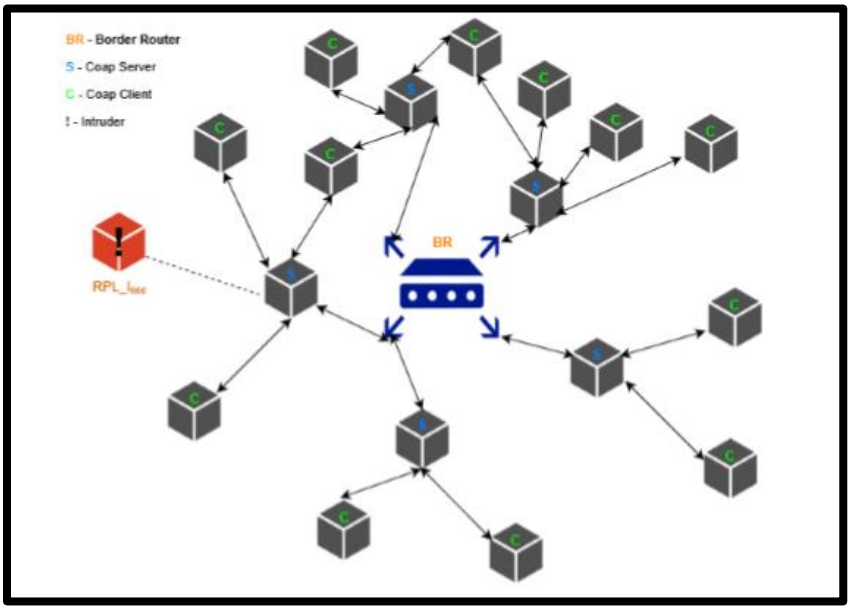


Figura 10:Esquema de sondagem para a solução proposta [10]

Os componentes da arquitetura são:

1) O Nó central 6BR que têm a função de conectar os demais a internet.

2) Os nós servidores que têm a função de disponibilizar a um grupo de clientes acesso a determinados recursos.

3) Os Nós clientes que acedem a recursos dos servidores, com uma taxa de solicitação temporizada.

Para o problema de deteção de intrusão temos particular interesse não apenas em saber se o sistema detetou ou não uma intrusão, mas também qual intrusão foi detetada pelo sistema.

3.3 Técnicas de aprendizagem e máquina aplicadas a internet das coisas

A inteligência artificial e a internet das coisas são dois temas atuais e promovem uma grande diversidade de objetos inteligentes com capacidade de processamento, sensoriamente, gerenciamento, interatividade, aprendizagem e Comunicação. Nas últimas décadas IoT e AI (Artificial Intelligence) têm

evoluído bastante, e têm sido aplicadas para facilitar a qualidade de vida das pessoas por meio de uma grande gama de aplicações, destinadas a industria, segurança, comercio, cidades, saúde etc., com estas tecnologias surgem questionamentos sobre a arquitetura, protocolos, softwares desenhados para estes ambientes entre outros, [22]**.**

A geração de falsos positivos e a eficácia do IDS são dois aspetos fulcrais para o sucesso da tarefa de deteção de intrusão em redes, razão pela qual os algoritmos de aprendizagem de máquina têm sido usados em muitas implementações na literatura para garantir a eficácia e reduzir o número de falsos positivos dos IDS [23]

Aprendizado de máquina: É um ramo da inteligência artificial focada no desenvolvimento de algoritmos com capacidade de aprender a resolver problemas sem intervenção humana direta [24], é um ramo interdisciplinar que envolve desde inteligência artificial, estatística e probabilidades, biologia, teoria da computação, entre outros mais diversos ramos da ciências

O aprendizado do algoritmo pode ser feito de três formas diferentes, de forma supervisionada, não supervisionada e por reforço.

Aprendizagem Supervisionada: Nesta abordagem o algoritmo tem como entrada um conjunto de valores de uma base de dados com determinados rótulos, para treinar um classificador que seja capaz de classificar novas instâncias não rotuladas [26]. Quando lidamos com valores discretos estamos diante de um problema de classificação, e quando estamos diante valores contínuos estamos diante de um problema de regressão [25].

Aprendizagem Não Supervisionada: Nesta abordagem o algoritmo recebe um conjunto de dados sem rótulos, e tenta extrair padrões destes dados, precisa-se de análise para perceber-se o significado destes padrões [25].

Aprendizagem por Reforço: Em situações em que é impraticável a aplicação de métodos supervisionados e não supervisionados, como por exemplo quando o agente vai operar em um ambiente desconhecido, esta abordagem pode ser útil pois nela o agente aprende com suas próprias experiências com o meio ambiente recebendo recompensas nos casos de decisões acertadas e punições nos casos contrários, com o tempo o agente deverá ser capaz de aprender quais as decisões tomar em cada situação específica, de formas a aumentar o número de recompensas e diminuir o número de punições [25].

Existem diversos algoritmos de aprendizagem de máquina descritos na literatura, entre os quais os mais frequentes para a tarefa de deteção de intrusão são: as máquinas de vetores de suporte SVM (Support Vector Machine), o algoritmo KNN (K-Nearest Neighbors), e as ANN (Artificial Neural Network), como exemplos os trabalhos desenvolvidos em : [23] [10].

SVM: As máquinas de vetores de suporte são amplamente usadas para a resolução de problemas de segurança, é usada para deteção de intrusão em vários estudos como em [26]. Este algoritmo constrói um hiperplano que divide o espaço de características em duas zonas, de modos a maximizar a margem de separação entre as mesmas e as novas instâncias são mapeadas para estas regiões e atribuídas a uma destas novas classes [27].

KNN: Este algoritmo aprende baseado em instâncias, ou seja, atribui a classe de novas instâncias a partir da classe maioritária obtida entre os seus vizinhos mais próximos do conjunto de treino [26]

ANN: As redes neuronais artificiais são baseadas no princípio de funcionamento do sistema nervoso biológico, as redes neuronais artificiais aproximam-se do sistema nervoso humano em dois aspetos: a rede adquire o conhecimento por meio de um processo de aprendizagem e o conhecimento é armazenado por meio de interconexões entre os neurônios [26].

Dentre os vários tipos de redes neuronais o algoritmo *perceptron* de uma única camada e *perceptron* multicamadas (MLP), são os mais usados em inúmeras aplicações na literatura [23], a figura 11 mostra os componentes fundamentais de um neurônio artificial.

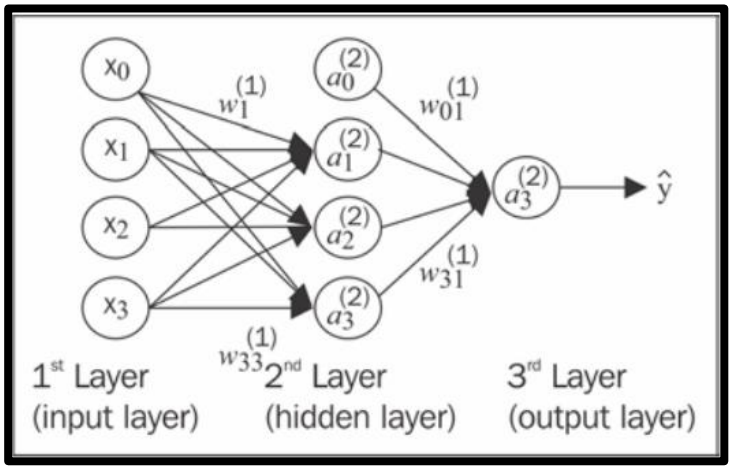


Figura 11:Arquitetura da rede neuronal artificial[3]

Na imagem acima podemos observar o princípio de funcionamento de um neurônio artificial, onde temos um conjunto de entradas, cada uma delas com pesos diferentes, estes valores darão entrada ao sistema que terá uma função de ativação para o neurônio da saída caso o valor de entrada for maior que um limiar [23].

Os pesos são chamados de *Sinapses*, aos pesos positivos chamamos de *sinapses* excitatórias e os pesos negativos são as sinapses inibidoras, os pesos aumentam ou reduzem o valor da entrada [30].

O algoritmo vai fazendo um ajuste de pesos até encontrar a combinação correta de pesos que se adeque aos valores da base de dados de treino [31].

Para o ajuste dos pesos é necessário o cálculo do erro, e este é feito através do algoritmo backpropagation. Este processo é feito em três etapas diferentes [31] [30].

- O valor de saída de cada neurônio pode ser calculado pela seguinte expressão:

$$U = \sum_{i=!}^{n} Wi * Xi$$

$$Y = g(u)$$

Onde:

$g(u)$: função de ativação

Wi : Pesos

Xi: Entradas

• De acordo a saída do sistema, calculamos o erro com uma função de custo, a função sigmóide é muito usada para este fim.

Função sigmóide : $f(x) = 1/(1 + e\hat{}x)$

O modelo vai sendo atualizado de acordo a derivada dos pesos.

- A propagação do erro é realizada, encontra-se a derivada de cada peso na rede e atualiza-se o modelo.

3.3.1 Trabalhos relacionados

Shahriari, Amini, Jalili em [28], apresentam uma proposta chamada de sistema RT-UNNID com redes neurais não supervisionadas para deteção de ataques conhecidos e não conhecidos no tráfego de rede em tempo real, usando o *dataset* KDD99. A deteção de intrusão foi realizada a partir de três aspectos diferentes: desempenho de deteção, tempo de treino e tempo de deteção, para detetar 27 tipos diferentes de ataque com três redes neuronais: a RT SOM, ART-1 e a ART-2, nesta experiência foi possível alcançar 95.74% para a SOM 97.421% para a ART-1 e 97.19% para a ART-2 de eficácia na deteção de ataques de DoS.

Liu, Yan, Wang em [29] apresentam uma abordagem híbrida voltada para WSN com redes neuronais, usando árvores de decisão e redes neuronais para deteção de ataques Dos. Nesta implementação toda a vez que as regras detetar uma anomalia, o pacote corrompido é encaminhado para um *perceptron* multicamadas com 50 neurônios na camada oculta e por meio de regras avalia-se se o pacote em causa representa ou não uma intrusão. Esta implementação também é feita sobre o dataset *KDD99* [37] .Esta implementação obteve uma taxa de deteção de 99.81% [29].

Semelhante à nossa pesquisa, um estudo comparativo é realizado em Almeida et al [30], onde são avaliados os desempenhos de duas RNA com o intuito de avaliar qual delas se adequa melhor a um IDS para detetar intrusões do tipo DoS em IoT. A comparação foi feita entre um *perceptron* multicamadas com pesos normais e um com pesos limitados. Esta RNAs foi implementada com 10 neurônios na camada oculta e o treino foi feito com o *dataset* KDD99. O Desenvolvimento foi feito em linguagem C, com o microcontrolador ARM Cortex-M3, Como resultados obteve-se 97% de accuracy usando somente 354 bytes de ROM (Read Only Memory) e 420 bytes de RAM (Random Acess Memory), os autores afirma que : durante o treino da ANN há uma resposta satisfatória do MLP (*multilayer perceptron*) alacançando uma boa accuracy para a deteção de ataques DoS [30].

Ghorbani e Lei em [31] implementam um IDS baseado em redes neuronais, este trabalho foi batizado com o nome de rede de aprendizagem competitiva melhorada. Nesta abordagem o peso do neurônio vencedor é atualizado com o valor mais próximo de entrada e os pesos dos neurônios perdedores são atualizados com valores mais distantes para a entrada. Para esta experiência utilizou-se o *dataSet* KDD99 que é por sinal o conunto de dados mais utilizado pelos pesquisadores a nível de deteção de intrusão e aprendizagem de máquina, disponível publicamente em [31].

Até ao momento podemos aferir que os trabalhos [28] [31] e [29], foram implementados sem levar em consideração as questões das restrições energéticas, de capacidade e processamento dos ambientes da internet das coisas.

O Trabalho desenvolvido em [30] não é descrita as etapas de seleção de características, e pré-processamento dos dados.

De realçar também que todos os trabalhos supracitados usam o dataSet KDD99, cujos alguns trabalhos apontam algumas limitações descritas em [32].

O trabalho desenvolvido em [23] , já leva em consideração as restrições energéticas dos dispositivos de internet das coisas, tendo como objetivo principal avaliar a viabilidade do uso de redes neuronais para estes dispositivos e estudar técnicas para a redução dos custos computacionais da RNAs usando o dataSet NSL-KDD disponível em : [48] a fim de treinar uma rede neural com 26 camadas na entrada, 9 camadas ocultas e 2 camadas de saída, usando um Arduíno ATmega328P de 8 bits com 32KB de memória flash e 2 KB de memória volátil.

Este estudo também investiga técnicas para a redução dos custos computacionais das redes neuronais para IoT, na sua experiência o autor afirma que mesmo não medindo o consumo de energia, a baixa potência máxima do microcontrolador presente no Arduíno UNO garantiu um baixo consumo de energia e alcança uma taxa de deteção de 97.14%

Segundo o autor o baixo custo computacional é garantido no artigo através de três maneiras:

- Seleção de características: Para redução do número de entradas e consequentemente do número de pesos.
- Utilização de apenas uma camada oculta, fundamentado em: [23]
- Por meio de experimentos garantir o menor número possível de neurônios na camada oculta sem que prejudique significativamente os resultados.

Embora que nesta pesquisa já é aplicável a dispositivos *Low-Power*, e tem a vantagem de usar o *dataSet* NSL-KDD, que é uma versão melhorada do KDD'99, usada por todos os outros pesquisadores, segundo [23] este conjunto de dados ainda sofre alguns problemas apresentados em [32], ele pode não representar perfeitamente de uma rede IoT real, mas devido a falta de conjuntos de dados disponíveis ao público para IDS são amplamente usados para pesquisas.

De realçar também que em nenhuma destas pesquisas foca-se em detetar intrusões em ambientes CoAP. Diferente do trabalho desenvolvido em [10], que se destina a detetar intrusão em IoT para ambientes CoAP e cujos os dados foram recolhidos de um ambiente propício para a simulação de redes de sensores sem fio o (IoT–LAB), implementado no sistema operativo que tem grande aceitação por parte da comunidade académica.

Esta pesquisa difere-se das demais porque visa desenvolver um novo classificador usando redes neuronais considerando as restrições de memória e processamento dos dispositivos de IoT, usando um conjunto de dados coletado de uma rede de internet das coisas real.

3.3.2 Avaliação do classificador

Para avaliar o nosso classificador usaremos métricas tais como: accuracy, taxa de verdadeiros positivos (True Positive Rate), taxa de verdadeiros negativos (True negative rate), Recall, precision e pontuação F1 (F1-Score) [4] [30] [24].

As métricas usadas para os problemas de deteção de intrusão normalmente estão ligadas a questões de classificação binária, pois o sistema deverá ser capaz de avaliar se uma ação é ou não uma intrusão do sistema. Desta feita as métricas que usamos para avaliar o desempenho do nosso sistema de deteção de intrusão, serão as mesmas usadas para um problema genérico de classificação binária [4].

Podemos observar a accuracy do classificador por meio da matriz de confusão que apresenta quatro valores possíveis.

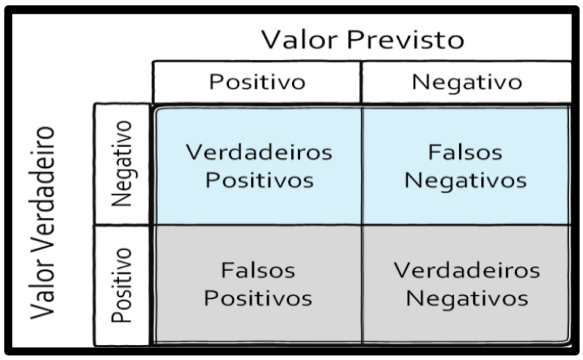


Figura 12:Exemplo Matriz de confusão[35]

Accuracy: Determina a qualidade do IDS em detetar corretamente todas as tentativas de intrusões ao sistema, e de não detetar erradamente ações não maliciosas, sendo definida pela razão dos resultados verdadeiros (verdadeiros positivos e verdadeiros negativos), e as demais atividades avaliadas pelo sistema. Esta métrica pode ser calculada pela seguinte expressão:

$$Exatidão = \ (TP + TN)/(TP + TN + FP + FN)$$

Precision: Esta métrica nos indica o número de ataques reais detetados pelo IDS corretamente, em relação ao número previsto de ataques reais, a ideia é conseguirmos alcançar a maior precision possível reduzindo assim a taxa de falsos positivos e pode ser calculada pela seguinte expressão:

$$precisão = TP/(TP + FP)$$

Recall: Avalia o grau de atividades maliciosas detetadas corretamente pelo IDS em relação ao número total de ataques reais. Está métrica pode ser calculada pela seguinte expressão:

$$TPR = TP/(TP + FN)$$

Taxa de falsos positivos (FPR): É a relação entre o número de ataques detetados incorretamente em relação ao número total de ataques reais.

$$FPR = FP/(FP + TN)$$

Taxa de falso negativos (TNR): É a proporção do número de ataques não reais não detetadas corretamente em relação ao número de ataques reais.

$$TNR = TN/(TN + FP)$$

Taxa de falsos negativos (FNR): É a relação entre o número de ataques reais não detetados em relação ao número total de ataques reais ao sistema.

$$FNR = FN/(FN + TP)$$

F1-Score: também chamada de F-measure é a média ponderada da recall e a precision. O seu valor varia entre [0,1], com 1 como seu melhor valor. Isto acontece quando a precision e recall são de 100%, sendo o cenário perfeito para IDS, com valor zero de falsos positivos e com uma taxa de deteção de 100%. [35] [30].

$$F1 - score = 2 * [(precisão * TPR)/(precisão + TPR)]$$

Curva ROC e AUC: As curvas das características de funcionamento do recetor (ROC) são úteis quando desejamos perceber a relação entre a taxa de verdadeiros positivos (TPR) e a taxa de falsos positivos (FPR), bem como para fins de comparação entre IDS. Ela mostra a taxa de falsos positivos (FPR) no o eixo das abcissas, e o TPR como no eixo das ordenadas. Uma vantagem em usar curvas de ROC é que estas possuem a capacidade de separar as considerações de custo de erro do desempenho do IDS, e uma desvantagem das mesmas consiste no facto de que pequenas variações na taxa de falsos positivos (FPR) afetam drasticamente a taxa de verdadeiros positivos (TPR), quando o tráfego normal excede em grande quantidade em comparação com intrusões na rede. No entanto existe uma outra métrica útil para comparar IDS a curva ROC (AUC) que avalia a precision total de uma forma que atende ao crescimento da taxa de TP e diminuição da taxa de falsos positivos. O seu valor fica entre 0,5(pior valor) e 1,0 (melhor valor) [30][4].

Capítulo 4

Implementação e Estratégia de Avaliação

Para a problemática da deteção de intrusão ao sistema duas abordagens foram seguidas nesta pesquisa, nomeadamente a abordagem binária e a múltipla. Na primeira abordagem pretende-se avaliar a capacidade do modelo de distinguir o tráfego normal representado no conjunto de dados pela **Label 0**, do tráfego intruso representado pela **Label 1**. Na classe múltipla pretende-se não apenas que o modelo distinga tráfego normal do anormal, mais também que o mesmo seja capaz de perceber qual tipo de intrusão foi detetada, considerando que num sistema de deteção baseado em anomalias os padrões dos dados de intrusão são previamente conhecidos, sendo que 4 comportamentos anormais estão presentes no conjunto de dados.

- *DOS FREQ* : Nó com uma taxa maior de solicitações de CoAP no respetivo servidor.
- *DOS ACK :* Envio de mensagens de confirmação, sem nenhuma solicitação enviada.
- *WRONG URI* : Solicitação de recursos não suportáveis por um nó.
- *WRONG ACCEPT* : Envio de solicitações como opção *ACCEPT* enviada para o servidor.

Estes comportamentos anómalos estão representados no conjunto de dados por labels distintas, organizadas da seguinte maneira:

DOS FREQ - **Label 1**

DOS ACK - **Label 2**

WRONG URI - **Label 3**

WRONG ACCEPT - **Label 4**

O conjunto de dados que vamos utilizar foi coletado da implementação de uma rede de internet das coisas sobre o protocolo CoAP, e durante os experimentos um conjunto de características (*features*) abaixo descritas, associados a rede de tráfego foram avaliados. O significado de cada uma destas *features* pode ser encontrado nos documentos oficiais dos protocolos IEEE 802.15.4 [39], 6LOWPAN [16] , e IPv6[38], e CoAP [19].

- wpan−nonask−phy.frame length - Frame Length
- wpan.aux sec.frame counter - Frame Counter
- wpan.bcn.gts.count - Guaranteed Time Slots (GTS) Descriptor Count
- wpan.cmd.gts.length - GTS Length
- wpan.correlation - Link Quality Indication (LQI) Correlation Value
- wpan.frame length - Frame Length
- wpan.gtsreq.length - GTS Length
- wpan.sec frame counter - Frame Counter
- wpan.sec key sequence counter - Key Sequence Counter
- 6lowpan.frag.size - Datagram size

- 6lowpan.fragment.count - Message fragment count
- 6lowpan.hc2.udp.length – Length
- 6lowpan.hops - Hop limit
- 6lowpan.iphc.hlim - Hop limit
- 6lowpan.mesh.hops - Hops left
- 6lowpan.nhc.ext.length - Header length
- 6lowpan.reassembled.length - Reassembled 6LoWPAN length
- 6lowpan.udp.length - Length
- ipv6.flow - Flow Label
- ipv6.fragment.count - Fragment count
- ipv6.hlim - Hop Limit
- ipv6.opt.calipso.cmpt.length - Compartment Length
- ipv6.opt.jumbo - Payload Length
- ipv6.opt.length - Length
- ipv6.opt.rpl.sender rank - Sender Rank
- ipv6.plen - Payload Length
- ipv6.reassembled.length - Reassembled IPv6 length
- ipv6.shim6.len - Length
- ipv6.shim6.opt.elemlen - Element Length
- ipv6.shim6.opt.len - Length
- ipv6.shim6.opt.total len - Total Length
- coap.opt.block size - Encoded Block Size
- coap.opt.length - Options Length
- coap.opt.length ext - Options extended Length
- coap.opt.max age - Max-age
- coap.token len - Token Length
- coap.code - Status Code

Desta feita passamos para a implementação de um algoritmo de classificação que seja capaz de detetar intrusões no conjunto de dados, para este efeito três fases fulcrais devem ser cumpridas como ilustra a figura 13. Na fase de pré-processamento os dados são preparados para o treinamento e como resultado teremos o modelo aprendido, usamos como ferramenta principal de trabalho a linguagem de programação *python* que é a linguagem de programação mais usada pelos cientistas de dados, por ser uma linguagem simples de aprender quando comparada com as outras, sua grande e crescente comunidade, seu vasto número de bibliotecas para ciência de dados tais como :*numpy, pandas, scikit-learn, keras*, sua escalabilidade bem como seu grande potencial para de visualização gráfica [40][41].

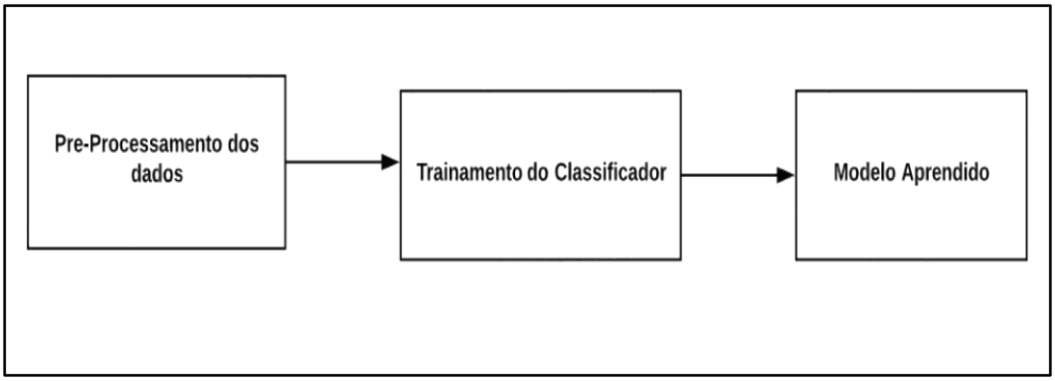


Figura 13: Etapas para a construção do modelo

4.1 Pré Processamento

O pré-processamento de dados é das fases mais importantes no processo de aprendizagem de máquina, visto que a eficácia dos algoritmos de aprendizagem de máquina depende grandemente da qualidade dos dados que forem submetidos para o treino [41].

Na fase de pré-processamento dos dados pretendemos melhorar a qualidade da informação que será submetida ao classificador, dado que em cenários reais as bases de dados apresentam diversos problemas que podem comprometer o desempenho dos algoritmos de aprendizagem de máquina [40], desde a presença de valores desconhecidos, presença de valores categóricos, elevada disparidade no número de exemplos para cada classe, presença de dados inconsistentes ou seja presença de valores que não façam sentido para os atributos que representam, ausência de dados, ausência de escalonamento dos atributos, presença de dados redundantes e outros[41][46].

Para a tarefa de pré-processamento dos dados prestamos particular atenção a presença de dados redundantes, pois pretendemos reduzir ao máximo o custo computacional do nosso modelo, o método *drop* da biblioteca *pandas* foi uma ferramenta útil para o cumprimento desta tarefa, bem como a etapa do escalonamento dos atributos, visto que a ausência de escalonamento fará com que o algoritmo de maior relevância a um atributo em detrimento de outros, o que pode comprometer a credibilidade dos resultados, e segundo [41] o escalonamento dos atributos melhora significativamente o desempenho dos algoritmos de aprendizagem de máquina. Para o cumprimento desta tarefa duas técnicas foram avaliadas, a normalização e a padronização, representadas pelas seguintes expressões matemáticas

Normalização

$$x = \frac{x - mínimo}{máximo(x) - mínimo(x)}$$

Padronização

$$x = \frac{x - média}{desviopadrão(x)}$$

De ambas selecionamos a padronização por ser mais robusta visto que leva em consideração o desvio padrão entre os dados, principalmente em conjunto de dados com *outliers* (observação dentro de um conjunto de dados que apresenta valores muito fora do comum e que podem comprometer a interpretação dos resultados) .

Para o cumprimento desta tarefa importamos a função *StandarScaler* da biblioteca *sklearn.preprocessing* e com o método *fit* efetuamos esta transformação nos atributos previsores, ou seja, nas variáveis independentes

4.2 Redução de Dimensionalidade

Bases de dados com vários atributos ou com uma grande dimensionalidade naturalmente demandam maior tempo para o treino e consequentemente maior custo computacional para o modelo [42][43], visto que adotamos como premissa termos um modelo com o menor custo computacional possível, técnicas para redução de dimensionalidade do conjunto de dados foram avaliadas , importa realçar que a redução de dimensionalidade não faz a seleção de características (escolher os atributos mais significativos dentro da base de dados), mais sim a extração de características (criação de novos atributos por meio dos já existentes, desta feita uma avaliação é feita sobre todos os atributos e novos atributos serão criados com base na correlação existente entre os atributos [41], para esta tarefa três abordagens foram avaliadas:

4.2.1 PCA (Principal Component Analysis)

A técnica de PCA ou análise das componentes principais, é um dos principais algoritmos não supervisionados para a redução de dimensionalidade, este algoritmo encontra os componentes principais também chamado de direções que maximizam a variância numa base de dados, tal como mostra a figura 14 [41] [43].

Caso tenhamos por exemplo **m** como número de atributos previsores (variáveis independentes), O algoritmo extrai **p <= m** novos atributos sem considerar a informação da classe, e por meio de testes escolhemos o valor mais adequado para P.

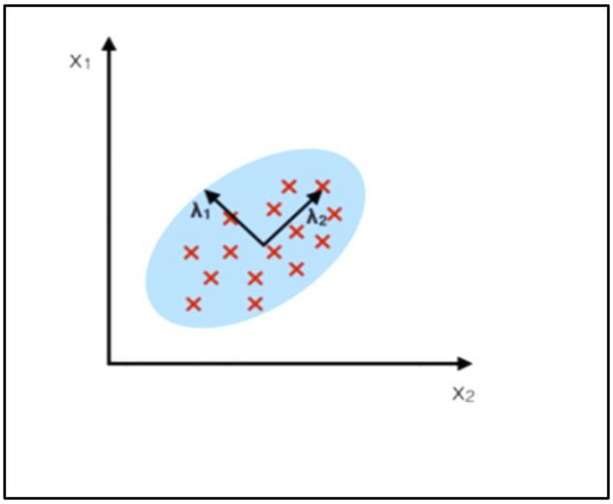


Figura 14:Componentes que maximizam a variância [41]

4.2.2 LDA (Linear Discriminant Analysis)

O LDA é outro algoritmo para redução de dimensionalidade, porém de aprendizagem supervisionada e nesta abordagem pretende-se encontrar os eixos que aumentam a separação entre as classes de modos a evitar o excesso de ajustes bem como reduzir o custo computacional do modelo tal como podemos observar na figura abaixo, dos **m** atributos previsores o algoritmo extrai **p <= m** variáveis independentes novas que sejam capazes de maximizar ao máximo as classes da variável dependente [41] [42].

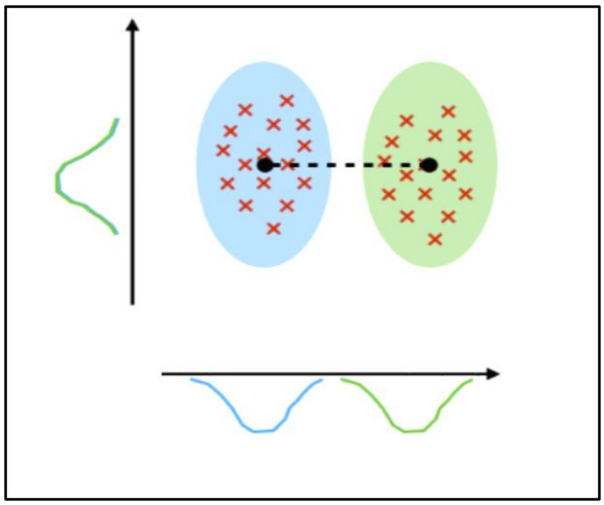


Figura 15:Maximizando os eixos dos componentes para a separação da classe[41]

4.2.3 Kernel PCA

As duas técnicas acima descritas são adequadas para problemas linearmente separáveis, e maior parte dos problemas mais complexos não são linearmente separáveis. *Kernel* PCA é uma variante do PCA adequada para problemas não linearmente separáveis onde os dados são mapeados para uma dimensão superior usando o conceito de *Kernel trick* (Uso de funções de *Kernel* por parte de um conjunto de algoritmos de classificação de padrões, dentre os quais o SVM, "permitindo o algoritmo a funcionar num espaço de recurso implícito de alta dimensão, sem necessidade de computar as coordenadas dos dados naquele espaço, mas simplesmente computando os produtos internos entre as imagens de todos os pares de dados no espaço de recurso)" [52], e dos componentes dos dados com maior , os componentes principais são extraídos como ilustra a imagem abaixo: [41] [43].

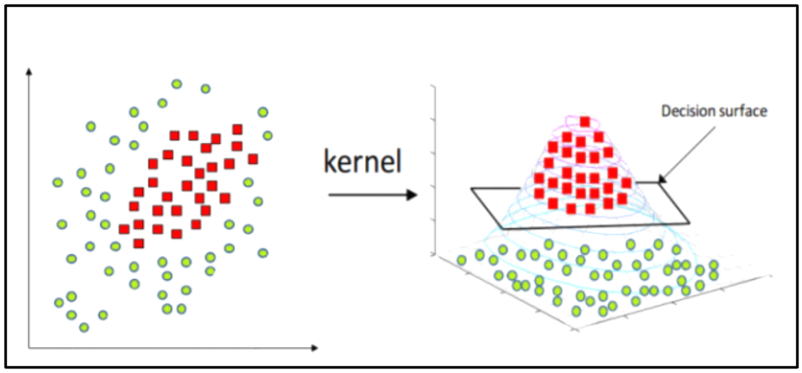


Figura 16: Kernel PCA [41]

Da biblioteca *sklearn.neural_network* [46], importamos a função *MLPClassifier* que é a nossa rede neural como tal e por meio desta criamos o nosso classificador configurando seus mais diversos parâmetros.

Alguns parâmetros mereceram a nossa atenção nesta avaliação pela influência que os mesmos têm sobre os resultados obtidos, desta feita avaliamos: O número de camadas ocultas, a função de ativação, A taxa de aprendizagem, e o *solver* usado.

Podemos aferir durante a pesquisa que é de consenso no meio da comunidade acadêmica definir-se 30 como o número de iterações para o resultado de uma avaliação de modos a garantir resultados mais confiáveis, o que replicamos neste estudo [10] [24].

4.3 Taxa de Aprendizagem:

A taxa de aprendizagem da rede neuronal é o parâmetro que define como os pesos da rede estão sendo ajustados em relação a descida do gradiente, Um valor baixo para a taxa de aprendizagem significa uma descida mais lenta do gradiente, o que é vantajoso por garantir que não perdemos nenhum mínimo local, mais por outro lado vai requerer muito mais tempo para convergir, caso queiramos um tempo de convergência menor naturalmente aumentamos a taxa de aprendizagem, sendo assim a melhor configuração deste parâmetro um valor que diminua ao máximo o tempo de treino do nosso algoritmo sem comprometer

o desempenho do sistema. A expressão matemática abaixo mostra a relação entre a atualização dos pesos, o gradiente e a taxa de aprendizagem [44] [45].

$$Novopeso = Pesoactual — Taxa\_de\_aprendizagem * gradiente$$

Podemos ver por meio da figura abaixo como diferentes configurações da taxa de aprendizagem podem se comportar graficamente

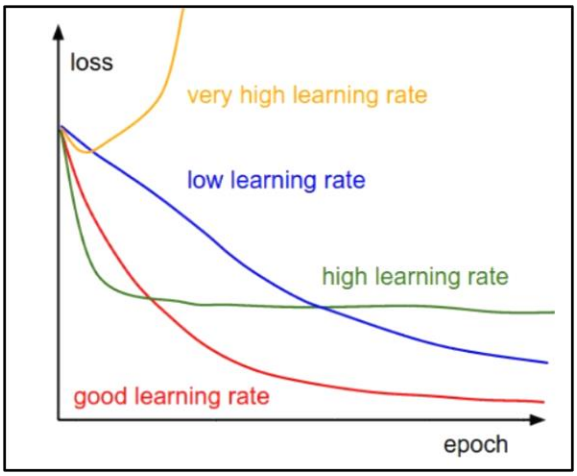


Figura 17:Comportamento de diferentes taxas de aprendizagem na convergência[45]

Para o caso do nosso modelo configuramos a taxa de aprendizagem começando com um valor pequeno e aumentamos gradualmente este valor em cada iteração [45]

4.4 Funções de Ativação:

As funções de ativação ou funções de transferência são usadas para que tenhamos o valor de saída de um neurônio da rede neural, dependendo da função usada o seu valor pode estar entre 0 e 1 ou entre -1 e 1, estas podem ser funções de ativação lineares ou não lineares.

A função linear tal como nos mostra a figura abaixo é basicamente uma linha reta, este tipo de função é apenas útil para problemas simples sendo que precisamos de funções mais robustas para problemas mais complexos [47].

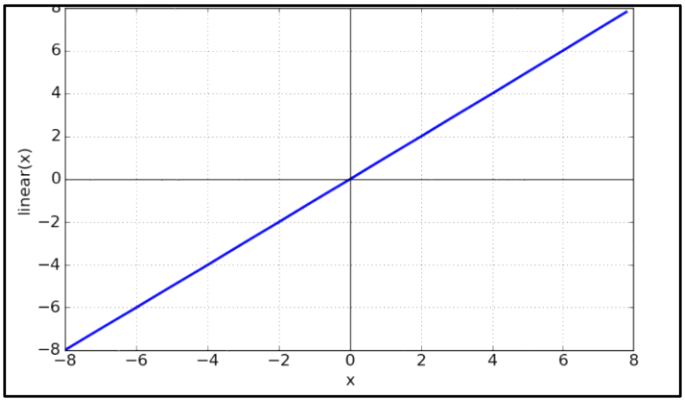


Figura 18:Função de ativação Linear [47]

Funções de ativação não lineares mostram-se mais eficazes para problemas mais complexos, durante as nossas avaliações testamos os resultados com 3 funções de ativação diferentes [46]

4.4.1 Função Sigmóide

A função sigmóide é das funções de ativação mais usadas e o valor da sua saída varia entre 0 e 1, sendo esta de grande utilidade para estudos probabilísticos [46]

A função sigmóide é uma função suave (contínua e diferenciável), esta função tem como desvantagem a possibilidade da rede neuronal ficar presa durante o treino segundo [46], a figura abaixo mostra-nos como é calculado o valor desta função e sua respetiva representação gráfica [47]

$$Y = \frac{1}{1 + e^{-x}}$$

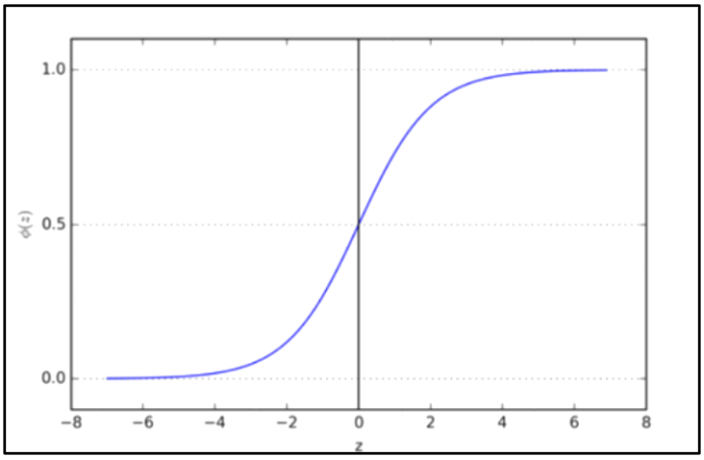


Figura 19:Representação gráfica da função Sigmóide[47]

4.4.2 Função tangente hiperbólica

A função tangente hiperbólica é uma função diferenciável, semelhante a função *sigmóide*, mais nesta podemos obter valores negativos na saída, visto que podemos ter valores entre -1 a 1, é uma alternativa mais robusta para ativação de camadas ocultas das redes neuronais. As duas funções de ativação vistas até agora são usadas em redes neuronais do tipo *feed-forward*, abaixo a expressão matemática e por meio da Figura 20 a representação gráfica de uma função tangente hiperbólica [46] [47].

$$y = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

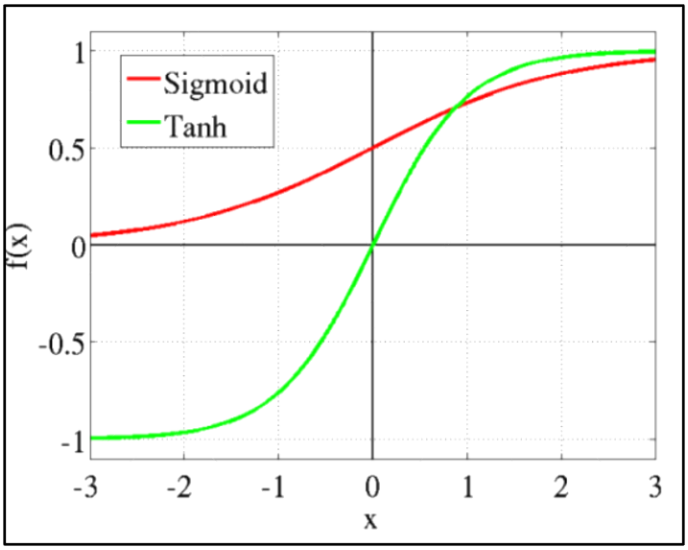


Figura 20:Representação gráfica da função tangente[47]

4.4.3 Função de ativação ReLU (unidade linear retificada)

Segundo [45] a função de ativação ReLU é a mais usada atualmente pelos pesquisadores principalmente para tarefas ligadas a *deep learning* (aprendizado profundo), é a função padrão para a biblioteca *scikit-learn.*

Tal como nos mostra a figura abaixa a função ReLU é meio retificada, ou melhor seus resultados encontram-se no intervalo de 0 até o infinito, abaixo a expressão matemática e a representação gráfica desta função:

$$\text{Y}= x^{+} = max(0, z)$$

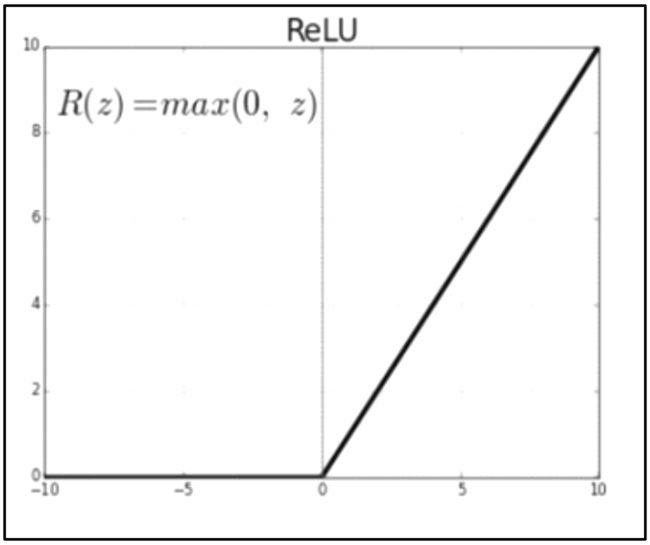


Figura 21:Representação gráfica da função ReLU [47]

4.5 Número de camadas Ocultas

No capítulo 3, falamos sobre do *perceptron* multicamadas, onde temos uma primeira camada, chamada de camada de entrada, a última é chamada de camada de saída e as camadas intermédias são chamadas camadas ocultas, para treinar uma rede neuronal *perceptron* multicamada geralmente usa-se o algoritmo de *backpropagation* [41].

A princípio quanto mais camadas e quanto maior o número de neurônios nas camadas ocultas melhor será o desempenho do nosso algoritmo, embora nem sempre o incremento de camadas ocultas e do número de neurônios melhora significativamente os resultados

Começamos a nossa avaliação sabendo a priori que é possível resolvermos problemas de deteção de intrusão em IoT com uma rede neuronal com uma única camada oculta [23], e fizemos testes sucessivos variando o número de neurônios nas camadas ocultas para testar até que ponto este parâmetro teria influência nos nossos resultados e qual o menor número possível de neurônios poderíamos utilizar para garantir um baixo custo computacional do modelo sem comprometer o desempenho do mesmo, começamos os testes definindo 23 como o número de neurônios na camada oculta visto que segundo pesquisas feitas em [46] , é possível resolvermos problemas de aprendizagem de máquina com redes neuronais definindo um número de camadas ocultas igual a metade do somatório entre o número de camadas de entradas e o número de camadas na saída, como abaixo enunciado pela expressão matemática:

Número de neurônios = Entrada + Saída /2

4.6. Solver

Solver: O solver é basicamente o algoritmo utilizado para a otimização dos pesos, a biblioteca *scikit-learn* comporta 3 tipos de solver e os descreve da seguinte maneira [46].

4.6.1 Adam

Refere-se ao optimizador substituto para a descida estocástica do gradiente, proposto por *Kingma, Diederik* muito usado no treino de modelos de aprendizagem profunda. O adam combina características dos algoritmos *AdaGrad,* (um algoritmo de otimização, proposto por Duchi, Hazan e Singer que adapta individualmente as taxas de aprendizado de todos os parâmetros do modelo escalando-os de forma inversamente proporcional à raiz quadrada da soma de todos os seus valores quadrados históricos e *RMSProp* (um algoritmo de otimização não publicado projetado para redes neurais, proposto pela primeira vez por Geoff Hinton), de modos a lidar com gradientes esparsos, é um solver simples a nível de configuração e apresenta bons resultados na maioria dos problemas, o adam funciona muito bem em conjuntos de dados relativamente grandes (com milhares de amostras de treino ou mais), e é *solver* padrão da biblioteca *scikit-learn* [46][47].

4.6.2 Sgd

É o s*olver* referente à descida do gradiente estocástico, é um algoritmo simples e eficiente e segundo apresenta bons resultados quando usado com algoritmos como as máquinas de vetores de suporte e regressão logística para tarefas de processamento de linguagem natural e classificação de texto, tem como vantagens a facilidade de implementação e sua eficiência e como desvantagens o facto de requerer vários hiperparâmetros [46].

4.6.3 Lbfgs

É um optimizador na família de métodos quase-Newton, é um algoritmo iterativo que minimiza uma função objetivo sujeita a algumas restrições de memória de uma matriz *bfgs* que aproximam a matriz hessiana no ponto em cada iteração [46] [50].

4.7 Validação Cruzada

Geralmente durante o desenvolvimento de um classificador dividimos o conjunto de dados em dois: um para treino e outro relativamente menor para testes. Esta forma de proceder é desvantajosa por ser possível que tenhamos registros na base de dados de testes que teriam maior relevância para o algoritmo se estivessem na base de dados de treino, e registros da base de dados de treino que seriam mais úteis se estivessem sendo usados para testes [41] [48].

A validação cruzada apresenta outra forma de divisão do conjunto de dados afim de colmatar esta lacuna usando todos os registros do conjunto de dados para testes e treino. Existem vários métodos para implementar a validação cruzada sendo a mais utilizada o método denominado *K-fold-cross-Validation* [51], nesta abordagem dividimos o conjuntos de dados em várias partes representadas pela constante k , geralmente um valor de *K =10* é usado em vários estudos[46] [45], sendo um valor bastante aceites pela comunidade científica, enquanto que uma *fold* está sendo usada para testes a outras serão usadas para treino e assim sucessivamente sendo que para um *K=10*, teremos 10 resultados em cada uma das divisões, sendo o resultado geral a média do resultado de cada divisão. Para implementar a validação cruzada avaliamos duas

funções da biblioteca *sklearn.model_selection* a_*Crossvalsocre* e a *StratifiedKFold*; A A função *StratifiedKFold* foi selecionada por implementar um método chamado de estratificação onde os dados são reorganizados de modos a garantir que cada um dos *folds* seja um bom representante do todo conjunto ou seja garantir que tenho uma boa distribuição entre as classes e por meio do parâmetro *Random_State* ir mudando a semente aleatória dos dados com o intuito de obtermos resultados diferentes em cada execução, o que garante uma avaliação mais eficiente [41].

4.7.1 Parâmetros importantes na validação cruzada

Para a efetivação da divisão estratificada foi necessário configurarmos de forma manual alguns parâmetros, tais como:

- *N_splits*: Por meio deste parâmetro definimos o valor de K, ou seja o número de divisões ou de *folds* , k igual a 10 foi o valor selecionado
- *Suffle*: defindo *true* garantimos a aleatoriedade da divisão dos dados
- *Random_State*: definindo o valor 1 para este parâmetro, configuramos a semente aleatória que vai fazer a divisão do conjunto de dados, isto garante que o conjunto de dados em cada teste será dividido em tamanhos diferentes para a obtenção de resultados distintos em cada avaliação, o que incrementa a confiabilidade dos resultados [49].

Como já acima mencionado foram feitas 30 iterações durante o treino, com a aplicação da validação cruzada teremos 10 resultados diferentes cada um deles com sua respetiva matriz de confusão, completando assim um total de 300 testes, a média de resultados dos testes gerais foi o valor considerado no final da avaliação. Os resultados obtidos serão agora analisados e debatidos no próximo capítulos

Capítulo 5

Resultados Obtidos e Discussão

Nesta fase do trabalho começamos a fazer testes sobre o modelo desenvolvido de modos a encontrarmos a melhor configuração para a nossa rede neuronal, sendo a melhor configuração aquela que demanda menor custo computacional sem comprometer o desempenho da rede para o cumprimento dos nossos objetivos.

5.1 Resultados Abordagem binária

Nesta seção apresentaremos os resultados obtidos na abordagem binária do nosso trabalho.

5.1.2 Redução de dimensionalidade

Começamos com o tópico da redução de dimensionalidade, pretendemos averiguar até que ponto o número de componentes principais afetará os nossos resultados, fazendo assim testes sucessivos com um número de componentes principais que varia de 3 a 9, apresentamos na tabela abaixo os resultados obtidos:

Parâmetros					**Resultados**			
NC	NN	FA	L.R	SV	Accuracy	Precision	Recall	F1-Score
3	23	ReLU	0.001	Adam	99.6%	50 %	50%	50%
6	23	ReLU	0.001	Adam	99.7%	50 %	50%	50%
7	23	ReLU	0.001	Adam	99.8%	100 %	100%	100%
8	23	ReLU	0.001	Adam	99.9%	100 %	100%	100%
9	23	ReLU	0.001	Adam	99.9%	100 %	100%	100%

Tabela 3:Resultado dos testes variando o número de componentes principais

Como podemos observar o número de componentes principais tem um impacto significativo sobre os resultados, naturalmente quanto maior o número de componentes teremos uma melhor representação do nosso conjunto de dados e desta feita melhoria significativa nos nossos resultados, não obstante maior demanda de recursos. Para 3 e 6 componentes obtivemos um valor de accuracy satisfatório, mais um valor baixo de precision, recall e F1-Score. É importante realçar que num conjunto de dados de intrusão existem muitos poucos exemplos de intrusão sendo que estes valores não são satisfatórios pois conseguem detetar apenas o tráfego normal e deixam passar os que de facto são intrusões ao sistema, a partir de 7 componentes principais começamos a obter resultados satisfatórios como nos mostra a Tabela 4.

5.1.2 Número de neurônios na camada oculta da rede neuronal

A tabela abaixo mostra-nos o resultado dos testes com um número diferente de neurónios na camada oculta da rede neuronal.

Parâmetros					Resultados			
NC	NN	FA	L.R	SV	Accuracy	Precison	Recall	F1-Score
3	5	ReLU	0.001	Adam	98.0%	50%	50%	50%
3	10	ReLU	0.001	Adam	98.2%	50%	50%	50%
3	15	ReLU	0.001	Adam	98.%	50%	50%	50%
3	23	ReLU	0.001	Adam	99.7%	100%	100%	100%
3	35	ReLU	0.001	Adam	99.8 %	100%	100%	100%

Tabela 4:Resultado dos testes variando o número de neurônios na camada oculta da rede neuronal

Quanto mais neurônios tivermos na camada oculta, melhor será o desempenho do nosso algoritmo, não obstante haja um incremento natural do custo computacional do modelo

Com 35 e 23 neurônios na camada oculta, obtemos resultados satisfatórios, e percebemos uma baixa de desempenho com valores abaixo de 23 neurônios na camada oculta, como podemos observar na Tabela 5.

5.1.3 Funções de ativação (ReLU, tanh sigmóide)

Depois de avaliarmos o comportamento do nosso modelo variando o numero de camadas ocultas da rede neuronal, vamos avaliar o impacto do nosso modelo usando diferentes funções de ativação, a tabela abaixo demostra os resultados dos testes:

Parâmetros					Resultados			
NC	NN	FA	LR	SV	Accuracy	Precison	Recall	F1-Score
3	23	Logistic	0.001	Adam	99.2 %	75 %	75%	75 %
3	23	Tanh	0.001	Adam	99.8 %	90 %	100 %	0,94 %
3	23	ReLU	0.001	Adam	99.8 %	100 %	100%	100 %

Tabela 5:Resultado dos testes variando a função de ativação

Sabemos que cada função de ativação tem suas valências em relação às outras, como podemos observar na tabela acima, começamos os nossos testes com a função *logistic* ou *sigmóide* e obtivemos um resultado satisfatório de accuracy, mais em algumas experiências o modelo não foi capaz de detetar as verdadeiras intrusões ao sistema, como podemos perceber pela imagem 11 da secção **c** em anexos, dando sequência aos testes com a função *tanh* os resultados melhoraram significativamente a nível de accuracy bem como para as demais métricas avaliadas , dando sequência aos testes obtivemos os melhores resultados para esta análise com a função ReLU, que por sinal é a função de ativação padrão do *scikit-learn* e a mais usada pelos pesquisadores, principalmente para tarefas de *deep learning* [44]. A vantagem da função ReLU é que nela não há saturação do gradiente, o que faz que esta seja mais rápida para convergir em comparação com as funções *sigmóide* e *tanh*, e por estas razões se mostra mais adequada para ser implementada em dispositivos com recursos computacionais limitados [53].

5.1.4 Avaliando o Solver

No teste anterior conseguimos perceber que as funções de ativação têm grande impacto sobre os resultados, vamos avaliar agora até que ponto o solver usado pode influenciar nos resultados obtidos, a Tabela 6 mostra os resultados desta avaliação:

Parâmetros					Resultados				
NC	NN	FA	L.R	SV	Accuracy	Precison	Recall	F1-Score	Tempo de treino
3	23	ReLU	0.001	Adam	99.6	100 %	100 %	100 %	40 min
3	23	ReLU	0.001	Sgd	99.6	100 %	100 %	100 %	40 min
3	23	ReLU	0.001	Lbfgs	99.8	100%	100 %	100 %	3 min

Tabela 6:Resultado dos testes variando o Solver

O solver *sgd* e o *adam* apresentam resultados iguais, já o *lbfgs* apresenta melhor desempenho e agrega outra mais valia pela redução do tempo de processamento. Em muitas implementações o solver *lbfgs* é um método de aproximação de quási-newton com memória limitada que mantem apenas aproximações simples de matrizes hessianas e consegue ter um bom desempenho de convergência geralmente não linear e, mostra-se adequada para os nossos interesses, embora o algoritmo que usaremos para treinar pode não ter impacto a nível de recursos porque não há necessidade de treinarmos o algoritmo no dispositivo, podemos muito bem treinar o algoritmo em um dispositivo com mais capacidade de processamento e memória e depois descarregar o modelo no esquipamento [50][46].

5.1.5 Avaliação com a taxa de aprendizagem

Neste derradeiro teste temos já definido o número de componentes principais a usar, o número de neurónios, a função de ativação e o *solver*, resta-nos perceber o impacto da variação da taxa de aprendizagem no nosso modelo, os resultados obtidos nesta avaliação podem ser observados por meio da tabela abaixo:

Parâmetros					Resultados			
NC	NN	FA	L.R	Solver	Accuracy	Precison	Recall	F1-Score
3	23	ReLU	0.001	Lbfgs	99.8 %	100%	100%	100%
3	23	ReLU	0.01	Lbfgs	99.8%	100%	100%	100%
3	23	ReLU	0.0001	Lbfgs	99.8%	100%	100%	100%

Tabela 7:Resultado dos testes variando a taxa de aprendizagem

A taxa de aprendizagem é parâmetro que não influencia os resultados obtidos, pois tal como afirma a documentação oficial do Scikit-learn este parâmetro tem relevância apenas quando usamos o solver *sgd* [46].

5.1. 6 Configuração final para abordagem 1

Depois dos testes realizados estamos em condições de definir a configuração final para o nosso modelo, considerando que a configuração ideal será aquela que temos a melhor relação desempenho/Custo computacional, estando esta descrita na tabela abaixo:

Parâmetros					Resultados			
NC	**NN**	**FA**	**LR**	**SV**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
3	23	ReLU	0.001	Lbfgs	99.8%	100%	100%	100%

Tabela 8:Configuração final para a abordagem binária

Outra métrica que mereceu a nossa atenção são as curvas de ROC (Receiver Operating Characteristic), usadas para avaliar a qualidade da saída do classificador, abaixo podemos ver o pior cenário possível para esta métrica (Auc_Score = 0,5), num dos casos de teste em que usamos apenas 10 neurônios na camada oculta, vê-se claramente que o modelo não consegue distinguir uma classe da outra considerando iguais o comportamento normal do sistema e intrusões.

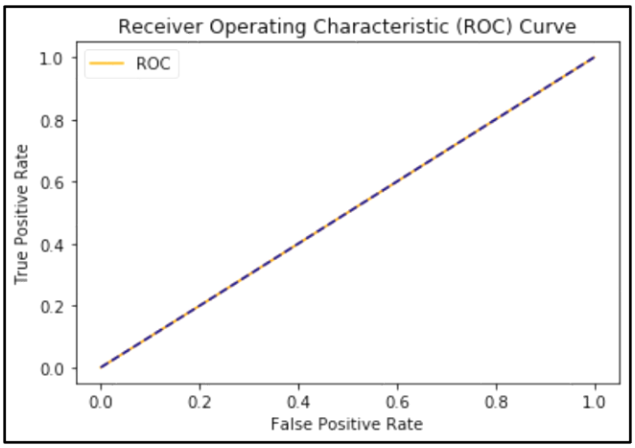


Figura 22:Curva de Roc do modelo usando 10 neurônios na camada oculta

A figura abaixo mostra-nos a curva de ROC dos parâmetros de configuração final descrita na Tabela 8, para a primeira abordagem e vê-se que o modelo consegue claramente distinguir uma classe da outra sendo este o melhor cenário possível para esta métrica (Auc _Score igual a 1).

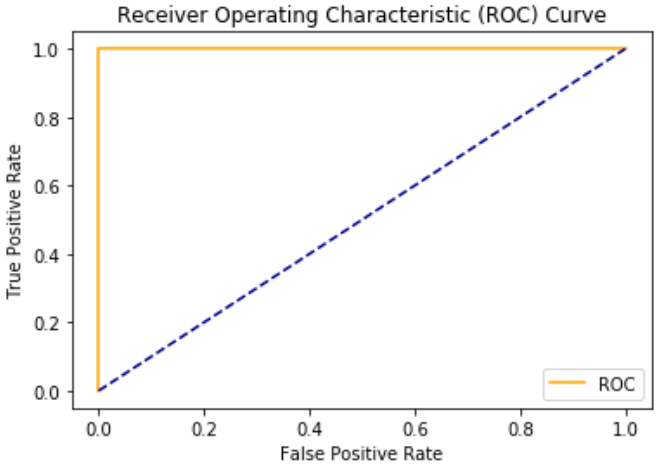


Figura 23: Curva de Roc abordagem 1

A matriz de confusão final para a parametrização final da primeira abordagem nos mostra que todos os dados foram classificados corretamente, obtendo assim uma taxa nula de alarmes falsos.

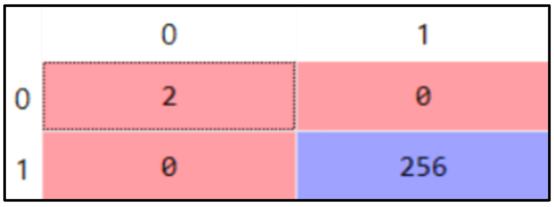


Figura 23: matriz de confusão Abordagem 1

5.2 Resultados Abordagem Múltipla

Nesta seção apresentaremos os resultados alcançados nos testes, para a abordagem múltipla

5.2.1. Componentes principais

Olhando para os resultados da tabela abaixo, podemos claramente observar que o número de componentes principais tem grande impacto sobre os resultados, ou seja quanto maior o número de componentes principais melhor será o desempenho do algoritmo.

Parâmetros					Resultados			
NC	NN	FA	L.R	SV	Accuracy	Precision	Recall	F1-Score
9	23	ReLU	0.001	Adam	98 %	97 %	96 %	96%
6	23	ReLU	0.001	Adam	97%	93%	96%	94 %
3	23	ReLU	0.001	Adam	95%	90%	91%	90 %

Tabela 9:Resultados da avaliação variando o número de componentes principais

5.2.1 Número de neurônios

A tabela abaixo nos mostra os resultados obtidos com diferentes números de neurônios na camada oculta, naturalmente quanto mais neurônios na camada oculta, melhores resultados obteremos.

Parâmetros					Resultados			
NC	NN	FA	L.R	SV	Accuracy	Precison	Recall	F1-Score
3	50	ReLU	0.001	Adam	98 %	98%	96%	97%
3	23	ReLU	0.001	Adam	97%	90%	91%	90%
3	15	ReLU	0.001	Adam	92 %	80%	77%	78%
3	5	ReLU	0.001	Adam	89 %	49%	60%	53%

Tabela 10:Resultado da avaliação variando o número de neurônios na camada oculta

5.2.3. Variando o Solver

Podemos ver abaixo que o solver com pior desempenho foi o *sgd e* o solver *lbfgs* e o ReLU alcançaram resultados melhores sendo que o *lbfgs* treina a rede neuronal em muito menos tempo tal como aconteceu na abordagem binária.

Parâmetros					Resultados			
NC	NN	FA	LR	SV	Accuracy	Precison	Recall	F1-Score
3	23	ReLU	0.001	Adam	97%	90	91	90
3	23	ReLU	0.001	sdg	94 %	75	73	74
3	23	ReLU	0.001	lbfgs	97%	90	91	90

Tabela 11:Resultado da avaliação variando o solver

5.2.4 Configuração final abordagem múltipla

Nesta abordagem já não fizemos testes com as funções de ativação pois concluímos que a *ReLU* é a mais adequada por meio dos experimentos feitos na primeira abordagem, nem com a taxa de aprendizagem visto que este parâmetro tem apenas impacto quando usamos o solver *sgd* , que por acaso nos apresentou os piores resultados em comparação aos demais. Com os resultados das avaliações estamos em condições de definir a configuração final para o nosso modelo, definimos a configuração ideal aquela que nos oferece a melhor relação desempenho/Custo computacional, a tabela abaixo mostra a configuração final para a abordagem múltipla.

Parâmetros					Resultados			
NC	**NN**	**FA**	**L.R**	**SV**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
3	50	ReLU	0.001	Lbfgs	98%	98 %	96%	97%

Tabela 12:Configuração Final abordagem múltipla

A matriz de confusão e a curvas características do recetor reforçam os resultados apresentados até agora, confirmando que maior parte dos casos foram classificados corretamente e que o modelo consegue distinguir perfeitamente o comportamento normal do sistema das intrusões, sendo que o classificador alcança um valor de Auc_Score igual a 1 para o ataque DOS FREQ e WRONG ACCEPT, 0.95 para DOS ACK e 0.94 para WRONG URI.

440.2	0.2	0.2	0	0.2
0.2	8.6	0	0	0
0.8	0.2	19.6	0	0
3	0	0	17.4	0
0.4	0	0	0	20

Figura 24:Matriz final abordagem múltipla

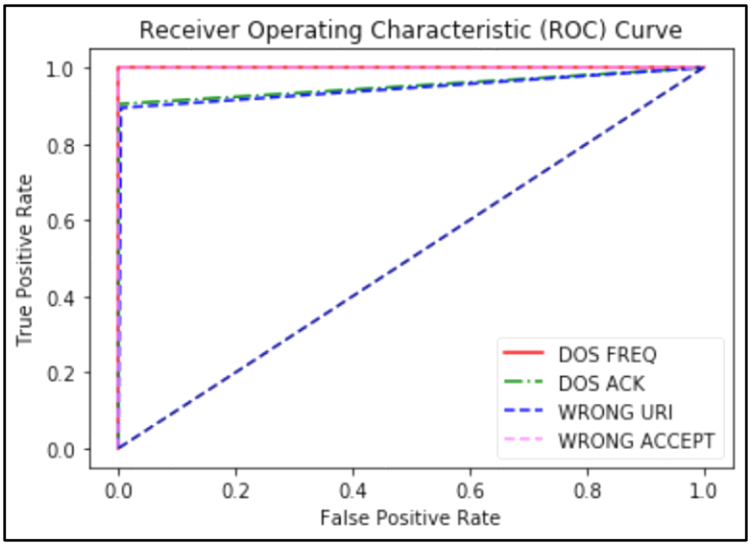


Figura 25:Curva de ROC para a abordagem múltipla

5.3 Conclusão do capítulo

Chegamos ao fim dos nossos testes e apresentamos a baixo uma breve comparação entre a implementação baseada em ANN e a implementação baseada em SVM

Na implementação com SVM foram obtidos resultados de 92% na abordagem binária e 93% para abordagem múltipla de accuracy e de 98% para Recall e F1_Score.

Neste trabalho de tese obtivemos resultados de 98% de accuracy, 98 % de precision, 96% e 97% de Recall e F1_score sucessivamente para a abordagem múltipla e de 99% de accuracy, 100% para precision, Recall e F1_Score para a abordagem binária.

Em ambos estudos foi possível aferir que com apenas 3 componentes principais é possível fazer uma classificação.

Para diminuir os custos computacionais dos modelos nas duas abordagens usamos técnicas de redução de dimensionalidade, LDA e PCA para a primeira implementação e Kernel PCA para a segunda pelo facto desta última ser mais adequada para problemas não linearmente separáveis.

A nível da modalidade de treino a implementação com SVM dividiu o conjunto em duas partes 70% para treino e 30% para os testes, neste estudo usamos a validação cruzada dividindo o conjunto de dados em 10 partes de modos a garantir maior fidelidade nos resultados.

As curvas de características do recetor para os dois modelos provam que ambos IDS conseguem distinguir claramente o tráfego normal das intrusões ao sistema.

Na primeira implementação as SVM foram selecionadas devido a sua rapidez de classificação e nesta as redes neuronais devido o seu grande poder computacional e desempenho na resolução de problemas complexos.

Capítulo 6

Conclusão e Trabalhos Futuros

Devido a problemática de se usar IPv6 em redes de baixa potência o IETF desenvolveu uma nova pilha protocolar dentre os quais o protocolo CoAP, 6LoWPAN e RLP, estes protocolos ajudam a interligar as WSN a internet. Esta conexão expõem as WSN a vulnerabilidades oriundas da internet por meio do protocolo IPv6 e para que a arquitetura atual de comunicações e segurança da internet cresça para suportar a IoT é importante garantir a sua segurança.

Sendo que os mecanismos de segurança tradicionais não são adequadas para ambientes IoT surge a necessidade de se desenvolverem sistemas de deteção de intrusão específicos para IoT e não existem atualmente IDS baseadas em anomalia de código aberto disponíveis para aplicações de segurança na internet das coisas, pelo que, nesta pesquisa desenvolvemos um classificador para deteção de intrusão baseado em anomalias com redes neuronais em semelhança ao desenvolvido com SVM presentes na biblioteca *Scikit-Learn* para a deteção de ataques de negação de serviço (DoS).

O classificador implementado foi avaliado com métricas genéricas de classificação binária, matrizes de confusão e curvas características de operação do recetor (ROC).

Pelos resultados alcançados durante a execução da tese podemos afirmar que:

Em cenários CoAP, o IDS é capaz de detetar comportamentos anómalos que são difíceis de detetar com IDS baseados em assinaturas.

Embora maior parte dos cientistas descarta a possibilidade de usar redes neuronais para aplicações de IoT, elas podem ser usadas em ambientes *LowPower* desde que seja avaliado o número de características e as restrições de memória e processamento do equipamento onde o modelo há de ser implementado.

Esta pesquisa mostra-nos um estudo cuidado das redes neuronais como alternativa as SVM para problemas de deteção de intrusão em IoT e apresenta melhores resultados.

Até esta etapa de investigação apenas as máquinas de vetores de suporte e as redes neuronais foram usados como classificadores e em estudos futuros outros algoritmos como k-NN, árvores de decisão, *Naive Bayes* e outros podem ser testados.

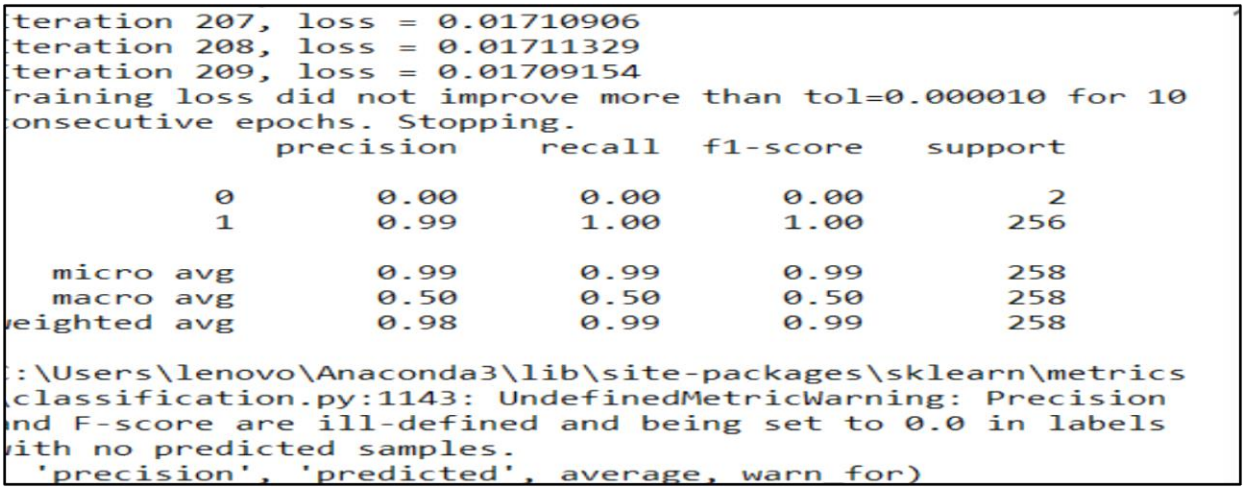
O modelo pode ser implementado em cenários reais de ataque, e novas intrusões podem ser testadas, visto que consideramos apenas os ataques de negação de serviço (DoS), as métricas de avaliação de desempenho tais como: CPU (Central Processing Unit), utilização da memória, e sobrecarga energética podem ser avaliadas, visto que os nós das redes IoT são geralmente alimentados por baterias.

Esperamos com esta pesquisa contribuir para a implementação bem-sucedida de ambientes de comunicação que permitam no futuro a integração de aplicações sensoriais ubíquas com a Internet, em particular ao nível da deteção de intrusões na presença de comunicações 6LoWPAN, RPL e CoAP de forma integrada.

Anexos

Secção 1 – Abordagem binária

Seção a) Imagens dos testes a nível da redução de dimensionalidade



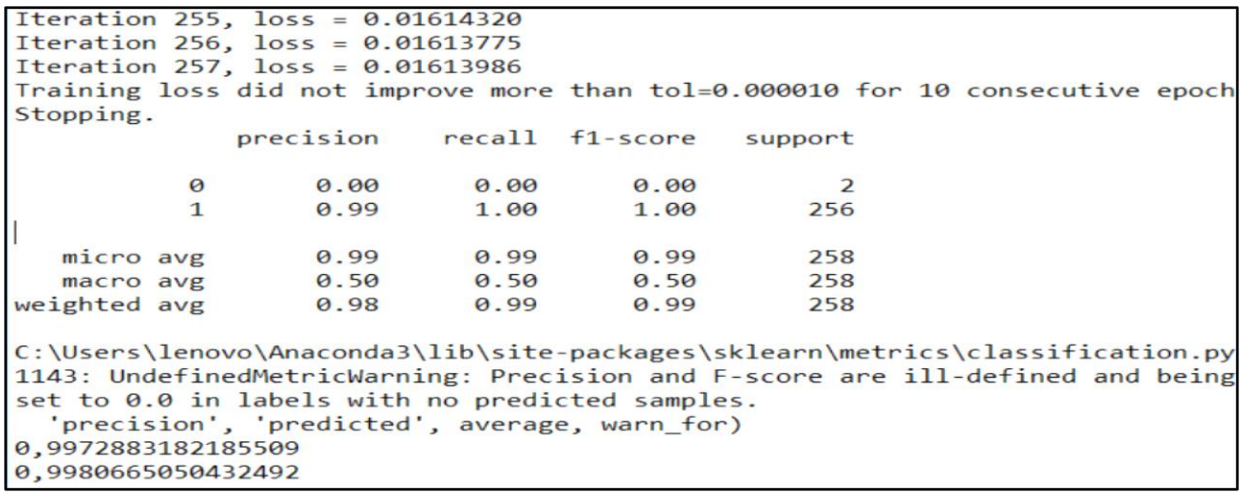
```
teration 207, loss = 0.01710906
teration 208, loss = 0.01711329
teration 209, loss = 0.01709154
raining loss did not improve more than tol=0.000010 for 10
onsecutive epochs. Stopping.
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         2
           1       0.99      1.00      1.00       256

   micro avg       0.99      0.99      0.99       258
   macro avg       0.50      0.50      0.50       258
eighted avg       0.98      0.99      0.99       258

:\Users\lenovo\Anaconda3\lib\site-packages\sklearn\metrics
\classification.py:1143: UndefinedMetricWarning: Precision
nd F-score are ill-defined and being set to 0.0 in labels
ith no predicted samples.
  'precision', 'predicted', average, warn_for)
```

Imagem 1:Redução de dimensionalidade para 3 componentes principais



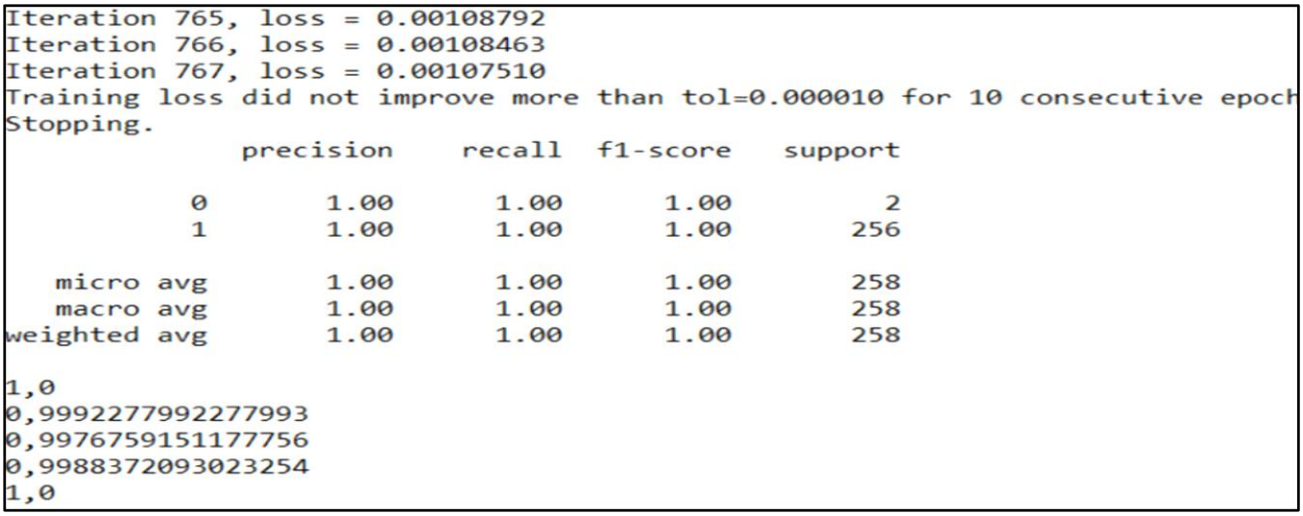
```
Iteration 255, loss = 0.01614320
Iteration 256, loss = 0.01613775
Iteration 257, loss = 0.01613986
Training loss did not improve more than tol=0.000010 for 10 consecutive epoch
Stopping.
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         2
           1       0.99      1.00      1.00       256

   micro avg       0.99      0.99      0.99       258
   macro avg       0.50      0.50      0.50       258
weighted avg       0.98      0.99      0.99       258

C:\Users\lenovo\Anaconda3\lib\site-packages\sklearn\metrics\classification.py
1143: UndefinedMetricWarning: Precision and F-score are ill-defined and being
set to 0.0 in labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
0,9972883182185509
0,9980665050432492
```

Imagem 2:Redução de dimensionalidade para 6 componentes principais



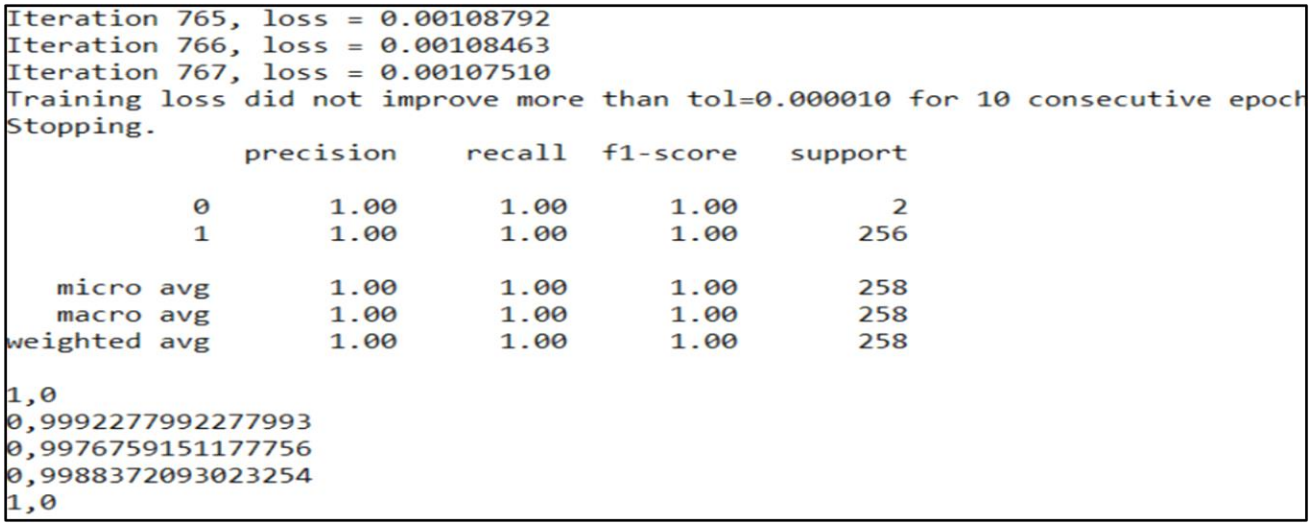
```
Iteration 765, loss = 0.00108792
Iteration 766, loss = 0.00108463
Iteration 767, loss = 0.00107510
Training loss did not improve more than tol=0.000010 for 10 consecutive epoch
Stopping.
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         2
           1       1.00      1.00      1.00       256

   micro avg       1.00      1.00      1.00       258
   macro avg       1.00      1.00      1.00       258
weighted avg       1.00      1.00      1.00       258

1,0
0,9992277992277993
0,9976759151177756
0,9988372093023254
1,0
```

Imagem 3:Redução de dimensionalidade para 7 componentes principais



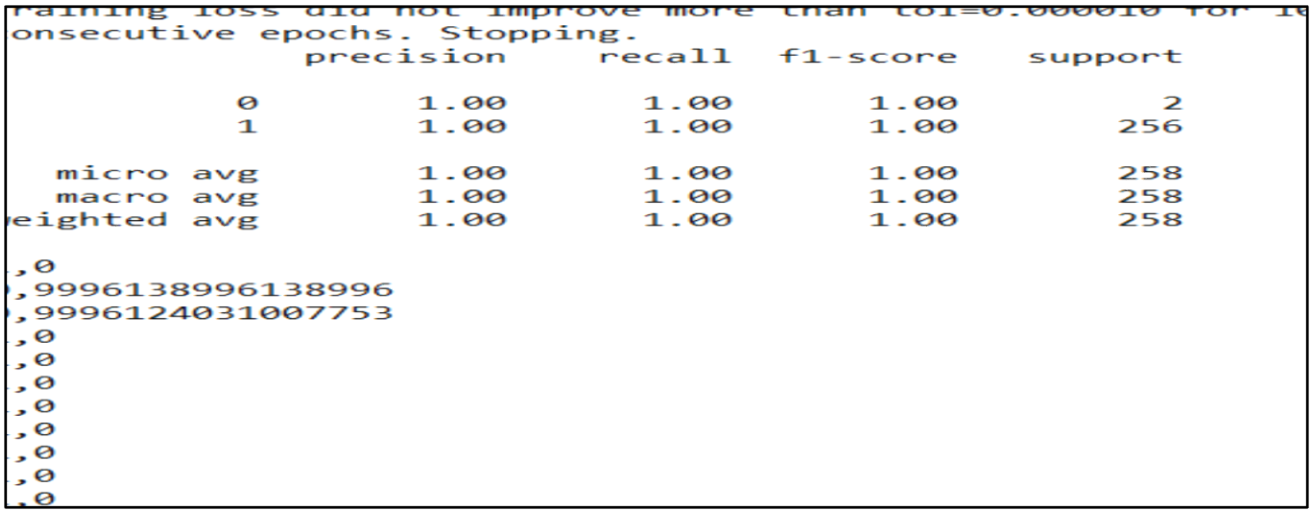
```
Iteration 765, loss = 0.00108792
Iteration 766, loss = 0.00108463
Iteration 767, loss = 0.00107510
Training loss did not improve more than tol=0.000010 for 10 consecutive epoch
Stopping.
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         2
           1       1.00      1.00      1.00       256

   micro avg       1.00      1.00      1.00       258
   macro avg       1.00      1.00      1.00       258
weighted avg       1.00      1.00      1.00       258

1,0
0,9992277992277993
0,9976759151177756
0,9988372093023254
1,0
```

Imagem 4:Redução de dimensionalidade para 8 componentes principais



```
raining loss did not improve more than tol=0.000010 for 1
onsecutive epochs. Stopping.
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         2
           1       1.00      1.00      1.00       256

   micro avg       1.00      1.00      1.00       258
   macro avg       1.00      1.00      1.00       258
eighted avg       1.00      1.00      1.00       258

,0
,9996138996138996
,9996124031007753
,0
,0
,0
,0
,0
,0
,0
,0
```

Imagem 5:Redução de dimensionalidade para 9 componentes principais

Seção b) Imagens de testes a nível do número de neurônios na camada oculta

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         3
           1       0.99      1.00      1.00       514

    accuracy                           0.99       517
   macro avg       0.50      0.50      0.50       517
eighted avg       0.99      0.99      0.99       517
```

Imagem 6:Resultados com 5 neurônios na camada oculta

```
UndefinedMetricWarning: Precision and F-score are ill-defined and being se
labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         2
           1       1.00      1.00      1.00       515

    accuracy                           1.00       517
   macro avg       0.50      0.50      0.50       517
weighted avg       0.99      1.00      0.99       517
```

Imagem 7:Resultados com 10 neurônios na camada oculta

```
Iteration 198, loss = 0.01658642
Training loss did not improve more than tol=0.000010 for 10 cons
Stopping.
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         2
           1       0.99      1.00      1.00       256

   micro avg       0.99      0.99      0.99       258
   macro avg       0.50      0.50      0.50       258
weighted avg       0.98      0.99      0.99       258
```

Imagem 8:Resultados com 15 neurônios na camada oculta

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         4
           1       1.00      1.00      1.00       512

    accuracy                           1.00       516
   macro avg       1.00      1.00      1.00       516
weighted avg       1.00      1.00      1.00       516

0,9984526112185688
0,9988394584139264
```

Imagem 9: Resultados dos testes com 23 neurônios na camada oculta

Secção c) Funções de ativação

```
           0       0.00      0.00      0.00         1
           1       1.00      1.00      1.00       516

    accuracy                           1.00       517
   macro avg       0.50      0.50      0.50       517
weighted avg       1.00      1.00      1.00       517

              precision    recall  f1-score   support

           0       1.00      1.00      1.00         1
           1       1.00      1.00      1.00       516
```

Imagem 10:Resultados dos testes com a função Sigmóide

```
    accuracy                           1.00       517
   macro avg       1.00      1.00      1.00       517
weighted avg       1.00      1.00      1.00       517

              precision    recall  f1-score   support

           0       0.80      1.00      0.89         4
           1       1.00      1.00      1.00       512

    accuracy                           1.00       516
   macro avg       0.90      1.00      0.94       516
weighted avg       1.00      1.00      1.00       516
```

Imagem 11:Resultados dos testes com a função Tanh

accuracy			1.00	516
macro avg	1.00	1.00	1.00	516
weighted avg	1.00	1.00	1.00	516

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1
1	1.00	1.00	1.00	516
accuracy			1.00	517
macro avg	1.00	1.00	1.00	517
weighted avg	1.00	1.00	1.00	517

Imagem 12:Resultados dos testes com a função ReLU

Seção d) Testes variando o Solver usado

	precision	recall	f1-score	support
0	1.00	1.00	1.00	2
1	1.00	1.00	1.00	515
micro avg	1.00	1.00	1.00	517
macro avg	1.00	1.00	1.00	517
weighted avg	1.00	1.00	1.00	517

Imagem 13:Resultados dos testes com o Solver adam

	precision	recall	f1-score	support
0	1.00	1.00	1.00	2
1	1.00	1.00	1.00	515
micro avg	1.00	1.00	1.00	517
macro avg	1.00	1.00	1.00	517
weighted avg	1.00	1.00	1.00	517

Imagem 14:Resultados dos testes com o Solver sgd

```
   macro avg       1.00      1.00      1.00       517
weighted avg       1.00      1.00      1.00       517

              precision    recall  f1-score   support

           0       1.00      1.00      1.00         4
           1       1.00      1.00      1.00       512

    accuracy                           1.00       516
   macro avg       1.00      1.00      1.00       516
weighted avg       1.00      1.00      1.00       516

1,0
0,9988394584139264
```

Imagem 15:Resultados dos testes com o Solver lbfgs

Seção e). Variando a taxa de aprendizagem

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         2
           1       1.00      1.00      1.00       515

   micro avg       1.00      1.00      1.00       517
   macro avg       1.00      1.00      1.00       517
weighted avg       1.00      1.00      1.00       517
```

Imagem 16:Taxa de aprendizagem de 0.001_0.01_0,0001

Secção 2 – Abordagem múltipla

Seção e) Testando com número de componentes principais

	precision	recall	f1-score	support
0	0.99	1.00	1.00	444
1	0.83	1.00	0.91	5
2	1.00	0.91	0.95	23
3	1.00	0.88	0.94	17
4	1.00	1.00	1.00	22
accuracy			0.99	511
macro avg	0.97	0.96	0.96	511
weighted avg	0.99	0.99	0.99	511

Imagem 17:Resultados dos testes com 9 componentes principais

	precision	recall	f1-score	support
0	0.99	1.00	1.00	449
1	0.67	1.00	0.80	4
2	1.00	0.91	0.95	23
3	1.00	0.88	0.93	16
4	1.00	1.00	1.00	19
accuracy			0.99	511
macro avg	0.93	0.96	0.94	511
weighted avg	0.99	0.99	0.99	511

Imagem 18:Testes com 6 componentes principais

```
Iteration 518, loss = 0.12233412
Iteration 519, loss = 0.12230770
Iteration 520, loss = 0.12232904
Iteration 521, loss = 0.12228866
Training loss did not improve more than tol=0.000010 for 10 consecutive
epochs. Stopping.
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       442
           1       0.69      0.90      0.78        10
           2       0.95      0.86      0.90        21
           3       1.00      0.89      0.94        19
           4       0.85      0.89      0.87        19

   micro avg       0.97      0.97      0.97       511
   macro avg       0.90      0.91      0.90       511
weighted avg       0.97      0.97      0.97       511
```

Imagem 19:Resultados dos testes com 3 componentes principais

Seção f) Teste número de neurônios nas camadas principais

```
              precision    recall  f1-score   support

           0       0.97      1.00      0.99       439
           1       0.00      0.00      0.00         8
           2       1.00      1.00      1.00        21
           3       0.00      0.00      0.00        24
           4       0.50      1.00      0.67        19

    accuracy                           0.94       511
   macro avg       0.49      0.60      0.53       511
weighted avg       0.89      0.94      0.91       511
```

Imagem 20:Resultado dos testes com 5 neurônios

```
labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       434
           1       0.00      0.00      0.00         6
           2       1.00      0.91      0.95        23
           3       1.00      0.95      0.98        22
           4       1.00      0.96      0.98        26

    accuracy                           0.98       511
   macro avg       0.80      0.77      0.78       511
weighted avg       0.97      0.98      0.97       511
```

Imagem 21:Resultado dos testes com 15 neurônios

```
Iteration 518, loss = 0.12233412
Iteration 519, loss = 0.12230770
Iteration 520, loss = 0.12232904
Iteration 521, loss = 0.12228866
Training loss did not improve more than tol=0.000010 for 10 consecutive
epochs. Stopping.
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       442
           1       0.69      0.90      0.78        10
           2       0.95      0.86      0.90        21
           3       1.00      0.89      0.94        19
           4       0.85      0.89      0.87        19

   micro avg       0.97      0.97      0.97       511
   macro avg       0.90      0.91      0.90       511
weighted avg       0.97      0.97      0.97       511
```

Imagem 22:Teste 23 neurônios

```
              precision    recall  f1-score   support

           0       0.99      1.00      0.99       439
           1       0.92      1.00      0.96        12
           2       1.00      0.90      0.95        21
           3       1.00      0.89      0.94        19
           4       1.00      1.00      1.00        20

    accuracy                           0.99       511
   macro avg       0.98      0.96      0.97       511
weighted avg       0.99      0.99      0.99       511
```

Imagem 23: Teste com 50 neurônios

Seção g) Testes com diferentes solvers

```
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       442
           1       0.75      0.90      0.82        10
           2       0.95      0.86      0.90        21
           3       1.00      0.89      0.94        19
           4       0.85      0.89      0.87        19

   micro avg       0.97      0.97      0.97       511
   macro avg       0.91      0.91      0.90       511
eighted avg       0.98      0.97      0.97       511
```

Imagem 24:Solver Adam e lbfgs

```
teration 1000, loss = 0.26929819
              precision    recall  f1-score   support

           0       0.96      0.99      0.98       442
           1       0.00      0.00      0.00        10
           2       0.95      0.86      0.90        21
           3       1.00      0.89      0.94        19
           4       0.85      0.89      0.87        19

   micro avg       0.96      0.96      0.96       511
   macro avg       0.75      0.73      0.74       511
eighted avg       0.94      0.96      0.95       511
```

Imagem 25:Utilizando O Solver SGD

```
In [30]: print(roc_auc_score(classe[indice_teste], previsoes))
1.0
```

Imagem 26:Resultado da roc_auc_score

7. Referências

[1] F. A. Fadele Ayotunde Alaba, Mazliza Othman a, Ibrahim Abaker Targio Hashema, "Internet of things Security: A Survey Fadele Ayotunde Alaba," *J. Netw. Comput. Appl.*, vol. 88, pp. 1–35, 2017.

[2] M. Henke, C. Santos, E. Nunan, E. Feitosa, E. dos Santos, and E. Souto, "Aprendizagem de máquina para segurança em redes de computadores: Métodos e aplicações," *Minicursos do XI SBSeg'2011*, vol. 1, no. September, pp. 53–103, 2011.

[3] P. Kasinathan, C. Pastrone, M. A. Spirito, and M. Vinkovits, "Denial-of-Service detection in 6LoWPAN based Internet of Things," *Int. Conf. Wirel. Mob. Comput. Netw. Commun.*, no. October, pp. 600–607, 2013.

[4] M. Dutta and J. Granjal, "Towards a Secure Internet of Things : A Comprehensive Study of Second line of Defense Mechanisms," 2020.

[5] L. M. R. Tarouco, I. J. Boesing, D. A. C. Barone, and G. R. P. Rosa, "Internet das Coisas na Educação trajetória para um campus inteligente," *An. dos Work. do VI Congr. Bras. Informática na Educ. (CBIE 2017)*, vol. 1, no. December, p. 1220, 2017.

[6] M. Nieles, K. Dempsey, and V. Y. Pillitteri, "An introduction to information security," 2017.

[7] W. D. E. Santo, E. Ordoñez, and A. Ribeiro, "Uma revisão sistemática sobre a Segurança nos Protocolos de Comunicação para Internet das Coisas," *J. Adv. Theor. Appl. Informatics*, vol. 4, no. 1, p. 1, 2018.

[8] J. Granjal, E. Monteiro, and J. S. Silva, "Security for the Internet of Things : A Survey of Existing Protocols and Open Research Issues," vol. 17, no. 3, pp. 1294–1312, 2015.

[9] S. Sakri, "Chapter 7 – Intrusion Detection and Security Analysis," *Nagios 3 Enterp. Netw. Monit.*, pp. 295–314, 2008.

[10] D. Report, "Attack and Intrusion Detection on the IoT," no. January, 2018.

[11] A. Gluhak, K. Srdjan, N. Michele, P. Dennis, M. Nathalie, and R. Tahiry, "T HE I NTERNET OF THINGS A Survey on Facilities for Experimental Internet of Things Research," vol. 11, no. November, pp. 58–67, 2011.

[12] U. S. Profile, "T H E D I G I T AL U N I V E R S E I N 2 0 2 0 : B i g D a t a , Bigger Digital Shadows , and Biggest Growth in the Far East — United States," *IDC iView IDC Anal. Futur.*, vol. 2007, pp. 1–7, 2013.

[13] E. Pietrosemoli, "Wireless standards for IoT: WiFi, BLE, SigFox, NB-IoT and LoRa."

[14] D. J. Brown, B. Suckow, and T. Wang, "A Survey of Intrusion Detection Systems 2 Information Sources Analysis Techniques."

[15] U. S. Unifacs, M. Em, and S. E. Computação, "Estudo Sobre Sistema De Detecção De Intrusão Por Anomalias : Estudo Sobre Sistema De Detecção De Intrusão Por Anomalias :," 2008.

[16] N. K. Gabriel Montenegro, Jonathan Hui, David Culler, "RFC 4944," 2007.

[17] M. C. da Silva, "Segurança em aplicações de redes de sensores com IPv6," 2013.

[18] T. Winter *et al.*, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks, RFC 6550," *Internet Engineering Task Force RFC 6550*. pp. 1–157, 2012.

[19] A. Shelby, Z, Track, and K. H. C. Bormann, "RFC 7252," p. 117, 2014.

[20] J. L. Luís ML Oliveira, Joel JPC Rodrigues, Amaro F Sousa, "No TitleDenial of service mitigation approach for ipv6-enabled smart object networks. Concurrency and Computation: Practice and Experience, 25(1):129–142," 2013.

[21] V. Vajda, K. Furdík, J. Glova, and T. Sabol, "The EBBITS Project : An Interoperability platform for a Real-world populated Internet of Things domain," *Int. Conf. Znalosti*, no. June 2014, pp. 309–312, 2011.

[22] W. Sousa, "ESTUDO DA INTELIGÊNCIA ARTIFICIAL APLICADA EM INTERNET DAS COISAS , VOLTADA NA AUTOMAÇÃO," PP. 1–29.

[23] F. de A. Florencio, E. D. Moreno, H. Macedo, R. J. P. de B. Salgueiro, F. B. do Nascimento, and F. A. O. Santos, "Intrusion Detection via Multilayer Perceptron using a Low-Power Device," pp. 1–5, 2019.

[24] R. Cerri, & A. de L. F.-C. de C., and U. 2019, "Aprendizado De Máquina: Breve Introdução E Aplicações," *Seer.Sct.Embrapa.Br*, pp. 297–313, 2019.

[25] D. Deshmukh and A. More, "International Journal of Innovative Research in Computer and Communication Engineering Applying Big Data in Higher Education," pp. 1301–1309, 2017.

[26] M. Henke, C. Santos, E. Nunan, E. Feitosa, E. dos Santos, and E. Souto, "Aprendizagem de máquina para segurança em redes de computadores: Métodos e aplicações," *Minicursos do XI SBSeg'2011*, vol. 1, pp. 53–103, 2011.

[27] T. Boeno *et al.*, "de água subterrânea Use of support vector machine for prediction of groundwater levels," pp. 25–34, 2018.

[28] M. Amini, R. Jalili, and H. R. Shahriari, "RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks," *Comput. Secur.*, vol. 25, no. 6, pp. 459–468, 2006.

[29] K. Q. Yan, S. C. Wang, and C. W. Liu, "A Hybrid Intrusion Detection System of Cluster-based Wireless Sensor Networks," *Proc. Int. MultiConference Eng. Comput. Sci.*, vol. I, 2010.

[30] F. Almeida, A. Ribeiro, E. Moreno, and C. Montesco, "Performance Evaluation of an Artificial Neural Network Multilayer Perceptron with Limited Weights for Detecting Denial of Service Attack on Internet of Things," *Twelfth Adv. Int. Conf. Telecommun. (AICT 2016)*, no. July, pp. 82–87, 2016.

[31] J. Z. Lei and A. A. Ghorbani, "Improved competitive learning neural networks for network intrusion and fraud detection," *Neurocomputing*, vol. 75, no. 1, pp. 135–145, 2012.

[32] H. Nkiama, S. Zainudeen, and M. Saidu, "A Subset Feature Elimination Mechanism for Intrusion Detection System," *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 4, pp. 148–157, 2016.

[33] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," *IEEE Symp. Comput. Intell. Secur. Def. Appl. CISDA 2009*, no. June 2014,

2009.

[34] Thingsquare, "Contiki: The Open Source OS for the Internet of Things," , 2014.

[35] S. Raschka and V. Mirjalili, "Python Machine Learning Second Edition Machine Learning and Deep Learning with Python, scikit-learn and TensorFlow," *John Wiley Sons, Inc.*, p. 309, 2006.

[36] O. Fambon, E. Fleury, G. Harter, R. Pissard-Gibollet, and F. Saint-Marcel, "FIT IoT-LAB tutorial: hands-on practice with a very large scale testbed tool for the Internet of Things," *UbiMob2014*, no. June, pp. 1–5, 2014.

[37] UCIKDD AchivedataSet , Disponível em: < http://kdd.ics.uci.edu/Acessado em junho (2019)

[38] Steve E. Deering and Robert M. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 8200, July 2017.

[39] Gabriel Montenegro, Jonathan Hui, David Culler, and Nandakishore Kushalnagar Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944, September (2007)

[40] Jake VanderPlas "Python Data Science Handbook", (2017)

[41] Sebastion Raschka ,Valid, Mirhalili "Machine learning and deep learning with python skit-learn and TensorFlow, Second Edition,Fully Revised and updated" (2007)

[42] GeoffreyJ.McLachlan"Discriminant Analysis and Statistical Pattern Recognition "(1992)

[43] I.T. Jolliffe "Principal Component Analysis", (2002)

[44] Leslie N. Smith U.S. Naval Research Laboratory, Code 5514 4555 Overlook Ave., SW., Washington "Cyclical Learning Rates for Training Neural Networks"(2018)

[45] Hafi dz Zulkifli, "Understanding Learning Rates and How It Improves Performance in Deep Learnning (2018) " Disponível em : https://towardsdatascience.com

[46] Tariq Rashid**,** Make your Own Neural Network: A gentle journey through the mathematics of neural networks, and making you own using the Python computer language

[47] SAGAR SHARMA **"**Understanding Activation Functions in Neural Networks" (2017) Disponível em https://medium.com/@avinashsharmav,Acessado em Dezembro de 2019

[48] Alice Zhen "Evalution Machine learning Models A beginners Guide for key concepts" (2013)

[49] Scikit-learn.org.https://scikit-learn.org/stable/modules/preprocessing.html (Acessado em Setembro de 2019

[50] Mokhtari, A .; Ribeiro, A."Convergência global de memória limitada online BFGS" (PDF) . Journal of Machine Learning Research . (2015)

[51] How to Correctly Validate Machine Learning Models https://rapidminer.com/resource/correct-model-validation/ (2019)

[52] *Theodoridis, Sergios* "*Reconhecimento de padrões"(2008)*

[53] Hypertext Transfer Protocol -- HTTP/1.1Disponivel em :http://www.w3.org/protocols/HTTP/1.1/rfc2616