UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
BACHARELADO CIÊNCIA DA COMPUTAÇÃO

Bernardo Brugnara Soccal

IDENTIFICAÇÃO DE COMPORTAMENTOS EM DISPOSITIVOS IOT UTILIZANDO MACHINE LEARNING

Santa Maria, RS
2019

Bernardo Brugnara Soccal

IDENTIFICAÇÃO DE COMPORTAMENTOS EM DISPOSITIVOS IOT UTILIZANDO MACHINE LEARNING

Trabalho de Conclusão de Curso apresentado ao Bacharelado Ciência da Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do grau de **Bacharel em Ciência da Computação**

Orientador: Prof. Dr. Raul Ceretta Nunes (UFSM)

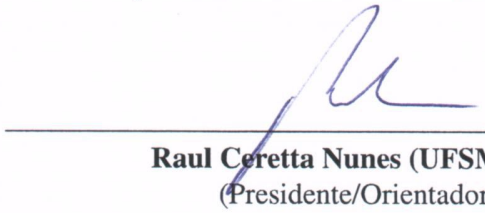
471
Santa Maria, RS
2019

Bernardo Brugnara Soccal

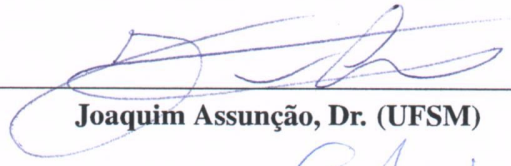
IDENTIFICAÇÃO DE COMPORTAMENTOS EM DISPOSITIVOS IOT UTILIZANDO MACHINE LEARNING

Trabalho de Conclusão de Curso apresentado ao Bacharelado Ciência da Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do grau de **Bacharel em Ciência da Computação**

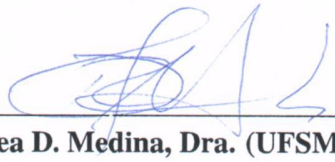
Aprovado em 12 de Dezembro de 2019:



Raul Ceretta Nunes (UFSM), Dr.
(Presidente/Orientador)



Joaquim Assunção, Dr. (UFSM)



Roseclea D. Medina, Dra. (UFSM)

Santa Maria, RS

2019

RESUMO

IDENTIFICAÇÃO DE COMPORTAMENTOS EM DISPOSITIVOS IOT UTILIZANDO MACHINE LEARNING

AUTOR: BERNARDO BRUGNARA SOCCAL
ORIENTADOR: RAUL CERETTA NUNES (UFSM)

Com o advento da Internet da Coisas (IoT), novos desafios surgiram na área de segurança de rede e identificação de dispositivos. A medida que novos dispositivos são introduzidos nos ecossistemas da IoT, se faz necessário ter um melhor controle da segurança e estrutura da rede, e para isso é preciso saber quais dispositivos estão sendo usados na rede e se estão sendo utilizados de maneira correta. Com uma grande variedade de tipos e modelos, muitos destes dispositivos apresentam vulnerabilidades exploráveis por agentes maliciosos, e para previnir possíveis ataques através destes dispositivos, é preciso saber quando os mesmos estão agindo de forma suspeita, o que só é possível detectar quando se tem o conhecimento do funcionamento padrão destes dispositivos.

Tendo isto em mente, neste trabalho são exploradas maneiras de identificar dispositivos IoT e diferenciar seus comportamentos entre quando estão agindo corretamente e quando estão apresentando comportamentos anômalos. Para isto são utilizadas duas bases de dados, uma contendo amostras de atividade de rede de 27 dispositivos IoT distintos, e outra com dados de tráfego botnet de dispositivos IoT infectados.

Usando os dados de cabeçalho dos pacotes transmitidos por estes dispositivos como base, algumas características foram escolhidas para a análise de Fingerprint dos dispositivos, que é o mapeamento de seus comportamentos em rede. Após montada uma base de dados contendo as características escolhidas, algoritmos de Machine Learning são utilizados como classificadores para identificar os dispositivos IoT, sendo estes algoritmos: Random Forest, Artificial Neural Network - Multilayer Perceptron, K-nearest Neighbors e Support Vector Machine.

Na identificação de dispositivos, o classificador que mostrou os melhores resultados foi o Random Forest, apresentando uma taxa de acertos variando entre 85% e 99%, sendo assim o algoritmo escolhido para fazer a diferenciação entre dados de dispositivos IoT normais e anômalos.

Os resultados finais foram bastante satisfatórios, com os classificadores tanto para dispositivos individuais, quanto para categorias de tipos de dispositivos IoT apresentando taxas de identificação correta entre dados normais e anômalos acima de 99%.

Palavras-chave: Internet das Coisas. Análise do tráfego de rede. Aprendizado de Máquina. Classificação de dispositivos.

ABSTRACT

BEHAVIOR IDENTIFICATION ON IOT DEVICES USING MACHINE LEARNING

AUTHOR: BERNARDO BRUGNARA SOCCAL
ADVISOR: RAUL CERETTA NUNES (UFSM)

With the advent of the Internet of Things (IoT), new challenges have emerged in the area of network security and device identification. As new devices are introduced into IoT ecosystems, better control of network security and structure are required, that is why you need to know which devices are being used on the network and whether they are being used correctly. With a wide variety of types and models, many of these devices have vulnerabilities exploitable by malicious agents, and to prevent possible attacks through these devices, you need to know when they are acting suspiciously, which you can only detect when the standard operation of these devices is known.

Keeping this in mind, this *Undergraduate Thesis* explores ways to identify IoT devices and differentiate their behaviors when they are acting correctly and when they are exhibiting anomalous behaviors. For this purpose two databases are used, one containing network samples of 27 separate IoT devices, and another with botnet traffic data from infected IoT devices.

Using the packet header data transmitted by these devices as a basis, some characteristics were chosen for the analysis of Device Fingerprint, which is the mapping of their network behaviors. After assembling a database containing the chosen characteristics, the following Machine Learning algorithms were used as classifiers to identify the IoT devices: Random Forest, Artificial Neural Network - Multilayer Perceptron, K-nearest Neighbors and Support Vector Machine.

In the identification of devices, the classifier that showed the best results was the Random Forest, presenting an accuracy rate varying between 85% and 99%, thus being the algorithm chosen to differentiate between normal and anomalous IoT device data.

The end results were quite satisfactory, with the classifiers for both individual devices and groups of IoT device types presenting correct identification rates between normal and anomalous data above 99%.

Keywords: Internet of Things. Network Traffic Analysis. Machine Learning. Device Classification.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

IoT *Internet of Things*

DFP *Device Fingerprinting*

ML *Machine Learning*

RF *Random Forest*

ANN *Artificial Neural Network*

MLP *Multilayer Perceptron*

SVM *Support Vector Machine*

KNN *K-nearest Neighbors*

SUMÁRIO

1 INTRODUÇÃO

A indústria de dispositivos da Internet das Coisas está crescendo em um nível extremamente rápido, havendo estimativas de que até 2020 existirão mais de 20 bilhões de dispositivos conectados globalmente. Estes dispositivos podem ser de tipos variados, como sensores, câmeras, termostatos, relógios, etc, e são em grande parte, destinados a melhorar a qualidade de vida das pessoas. No entanto, seja por concorrência de mercado ou limitações técnicas, muita vezes a segurança dos dispositivos acaba ficando em segundo plano (SENRIO, 2016). A falta de segurança nestes dispositivos pode acabar levando a vulnerabilidades exploráveis por agentes maliciosos. A infecção de um simples dispositivo IoT pode levar ao comprometimento de toda uma estrutura de rede (SENRIO, 2018).

Fazer a identificação do comportamento dos dispositivos conectados à rede é algo cada vez mais importante devido ao crescimento do número de dispositivos IoT espalhados pelo mundo, mas que ainda é muito negligenciado por diversas empresas (SENRIO, 2019).

De acordo com (WANG, 2013), é comum para mecanismos de segurança fazerem uso de identificação e classificação como parte central de seus sistemas (e.g., Firewalls e IDS - Sistemas de detecção de intrusão), com o objetivo de previnir atividades maliciosas. No caso da IoT, é importante fazer a identificação de dispositivos para que eles possam ser classificados de acordo com o nível de risco que os mesmo representam para a rede.

Atualmente, a fim de melhorar a segurança da rede e verificar o comportamento dos dispositivos IoT, técnicas de identificação de comportamento, como a Device Fingerprinting (DFP) (ANEJA; ANEJA; ISLAM, 2018) podem ser aplicadas, onde se tenta identificar dispositivos a partir de amostras da atividade de rede dos mesmos (BEZAWADA et al., 2018).

Ter acesso aos dados de atividade e comportamento dos dispositivos IoT conectados à rede possibilita fazer o treinamento de classificadores de Machine Learning (ML), os quais podem ser usados para identificar o tipo de dispositivo de acordo com as características de seu comportamento. Ter a capacidade de identificar os dispositivos conectados a uma rede proporciona um melhor controle de tráfego da mesma, e permite lidar melhor com dispositivos considerados um risco à segurança da estrutura da rede como um todo.

Além disso, conhecer o comportamento normal dos dispositivos aumenta as chances de se detectar quando os mesmos estão agindo fora do padrão, o que é essencial para evitar possíveis ataques a dispositivos vulneráveis.

1.1 OBJETIVO GERAL

O objetivo deste trabalho é identificar dispositivos IoT e diferenciar comportamentos normais e anômalos, analisando seus padrões de funcionamento através de amostras de tráfego de rede dos mesmos. Para este fim, são estudadas quais características de tráfego de rede dos dispositivos devem ser escolhidas para que a identificação do comportamento dos dispositivos seja a mais precisa possível.

Após escolhidas as características mais relevantes, bem como uma base de dados com amostras do tráfego de rede de 27 dispositivos IoT, é feito o treinamento de classificadores binários de *Machine Learning (ML)* para cada dispositivo, utilizando os algoritmos RF, MLP, KNN e SVM.

Com a conclusão dos testes, é realizado o ranqueamento dos classificadores de ML no que diz respeito a sua adequação em fazer a classificação entre dados comuns de dispositivos IoT e dados anômalos de dispositivos infectados por agentes maliciosos.

1.2 OBJETIVOS ESPECÍFICOS

- Aprofundar os conhecimentos na área de IoT, mais especificamente, no reconhecimento de comportamentos de dispositivos IoT;
- Reunir um compilado de literaturas sobre os tópicos deste trabalho;
- Implementar algoritmos de Machine Learning com um alto nível de acerto nas classificações de dados de dispositivos IoT;
- Descobrir quais algoritmos e caracteristicas de tráfego de rede são mais adequados para classificar o comportamento de dispositivos IoT.

1.3 ORGANIZAÇÃO DO TEXTO

O restante do trabalho está organizado da seguinte forma: no capítulo 2 é apresentada a fundamentação teórica dos principais conceitos usados no trabalho, o capítulo 3 contém a revisão bibliográfica, com os artigos relacionados que mais influenciaram este trabalho, nos capítulos 4, 5 e 6 são apresentadas as atividades práticas realizadas e os resultados obtidos, por fim, no capítulo 7 estão as considerações finais do autor.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo descreve os principais conceitos e algoritmos utilizados neste trabalho, com a seção 2.1 fazendo uma breve introdução a *Internet das Coisas (IoT)* e a seção 2.2 descrevendo a estrutura básica dos algoritmos escolhidos como classificadores de *Machine Learning*.

2.1 INTERNET DAS COISAS

A Internet das coisas (IoT) é, resumidamente, uma extensão da internet atual. Ela surgiu a partir da evolução de diversas áreas, como sistemas embarcados, microeletrônica, comunicação e sensoriamento, e proporciona acesso à internet aos objetos que fazem parte dela.

A IoT tem alterado ao longo dos últimos anos o conceito de "Redes de computadores", é possível perceber a evolução do termo ao observar citações como a de Tanenbaum (TANENBAUM; WETHERALL, 2002), “Rede de Computadores é um conjunto de computadores autônomos interconectados por uma única tecnologia”, sendo que atualmente os dispositivos são interconectados por diversos tipos de tecnologias diferentes, dentre as quais estão a Ethernet, WiFi, 4G, Bluetooth, dentre diversas outras tecnologias de comunicação (SANTOS et al., 2016). Definições como a de Peterson (PETERSON; DAVIE, 2011), que afirma que a principal característica das Redes de Computadores é a sua generalidade, não sendo elas otimizadas para fins específicos tais como as redes de telefonia e TV já não são mais tão validas atualmente, com a quantidade de dispositivos IoT projetados e otimizados para tarefas bem especificas.

Para Kurose e Ross (KUROSE; ROSS, 2012) o termo “Redes de Computadores” está ficando ultrapassado, tendo em vista a diversificação de equipamentos e tecnologias utilizados na internet. Atualmente, a quantidade de dispositivos conectados à internet é imensa, sendo que boa parte deste número se deve à dispositivos IoT, com previsões de que cerca de 45% de todo o tráfego de rede mundial será relacionado a IoT até 2022 (GARRETT et al., 2018).

Com uma grande variedade de dispositivos, incluindo sensores, câmeras, relógios e diversos tipos de aparelhos *smart*, a rede IoT só tende a aumentar nos próximos anos, trazendo com ela uma nova rede mundial de dispositivos.

2.2 MACHINE LEARNING E ALGORITMOS

Machine Learning (ML) é a ciência de programar computadores para que eles possam aprender com dados (GERON, 2019). Os classificadores de ML mais populares são projetados para aprenderem padrões com os dados de um problema, para conseguir chegar ao resultado mais provável.

Neste trabalho são testados 4 algoritmos de ML, para verificar qual deles consegue aprender melhor com os dados disponíveis. A seguir a estrutura básica destes algoritmos é apresentada.

2.2.1 Random Forest

Para falar de Random Forest primeiro é preciso apresentar a *Decision Tree (DT)*. Este algoritmo ao ler e aprender os padrões de uma base de dados monta uma estrutura no formato de árvore, com o nó raiz como ponto inicial e os nós folhas sendo as diferentes classes ao qual um dado sendo classificado pode pertencer. A imagem 2.1 mostra a estrutura de uma *Decision Tree* que prevê se o risco de empréstimo será Alto, Moderado ou Baixo de acordo com os atributos: História de crédito, Dívida e Renda anual (ex: 15 = R$ 15,000).

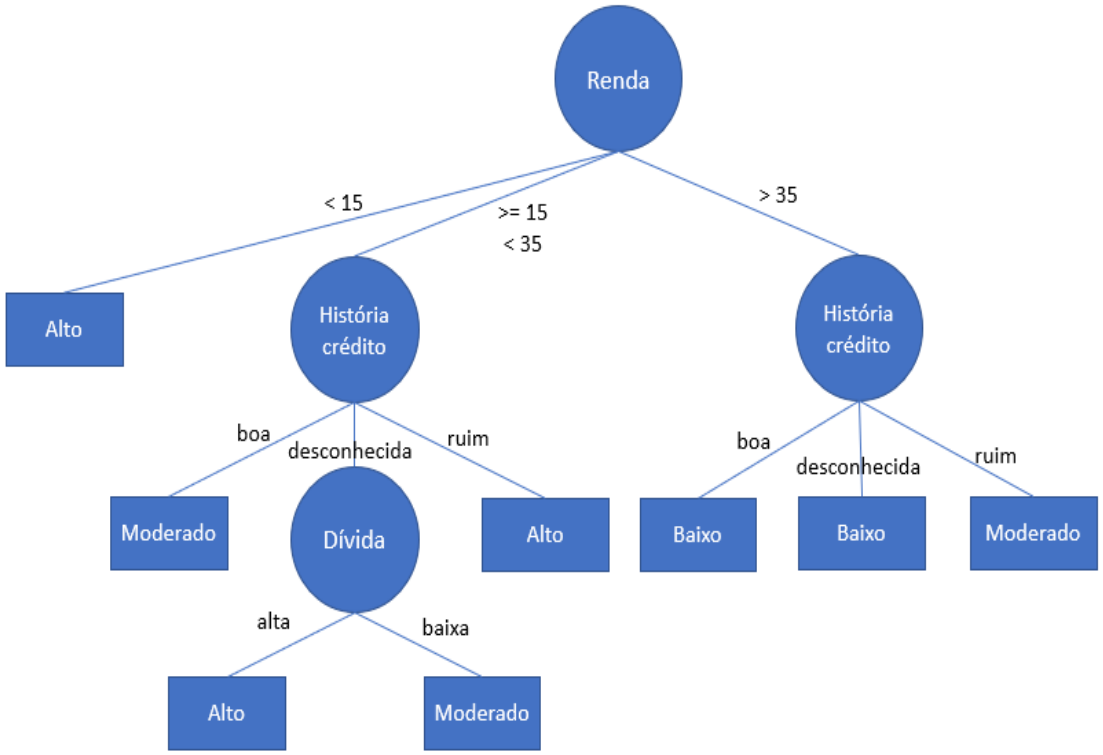


Figura 2.1: Decision Tree. Fonte: Curso Machine Learning e Data Science com Python de A a Z. Autor: Jones Granatyr

O Random Forest propriamente dito é uma melhoria sobre a DT, ele é um algoritmo do tipo *Ensemble Learning*, ou aprendizado em conjunto, onde mais de uma DT é utilizada como classificador, sendo que cada uma delas é treinada levemente diferente, gerando assim resultados distintos. No fim, o resultado é uma combinação dos resultados do conjunto de DTs, que no caso de classificadores é o resultado de uma votação da maioria. A imagem 2.2 ilustra o seu funcionamento básico, com X sendo a base de treinamento, e cada DT apresentando sua resposta que contribuirá para o resultado final.

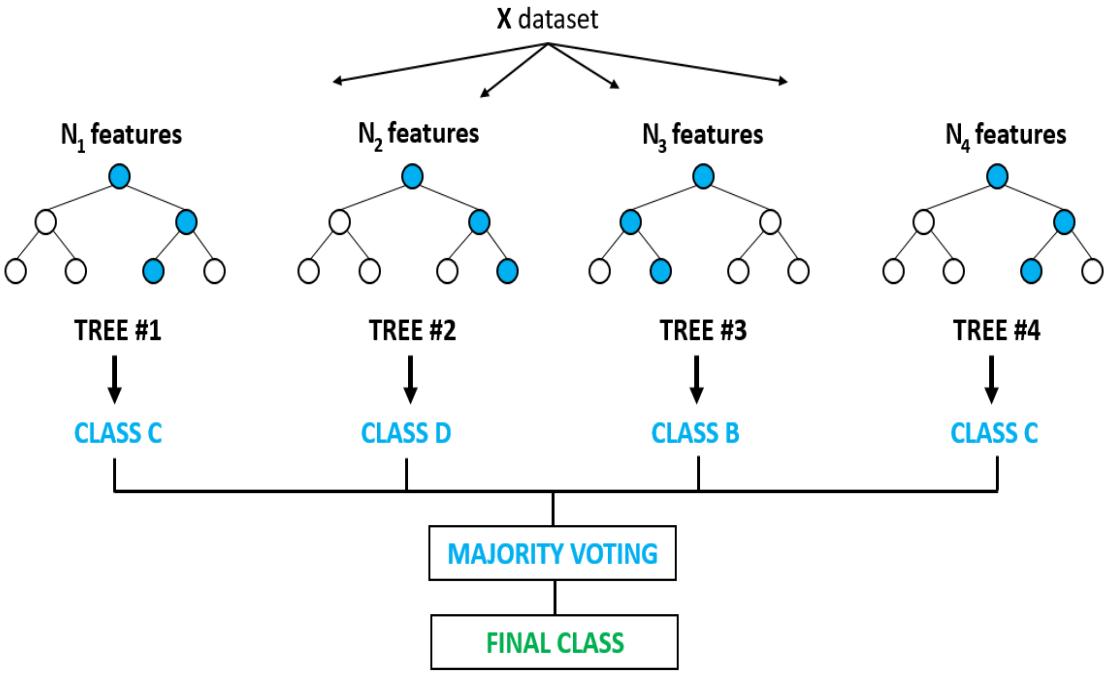


Figura 2.2: Random Forest. Fonte: GlobalSoftware[1]

2.2.2 Support Vector Machine

O SVM é um algoritmo de ML bastante versátil, capaz de executar classificação linear e não linear. Seu funcionamento básico se dá através da aprendizagem de hiperplanos com margem máxima, onde uma linha é traçada entre duas classes de dados, sendo que o objetivo é que esta linha esteja o mais longe possível da instância mais próxima de cada classe. A imagem 2.3 ilustra o funcionamento do SVM linear, onde um linha reta pode ser traçada para separar as duas classes. Nesta imagem, as classes *star* e *circle* são separadas pela linha vermelha, e a margem máxima (Gap) é delimitada pelas instâncias mais próximas, marcadas em amarelo.

[1]Link:https://www.globalsoftwaresupport.com/random-forest-classifier/

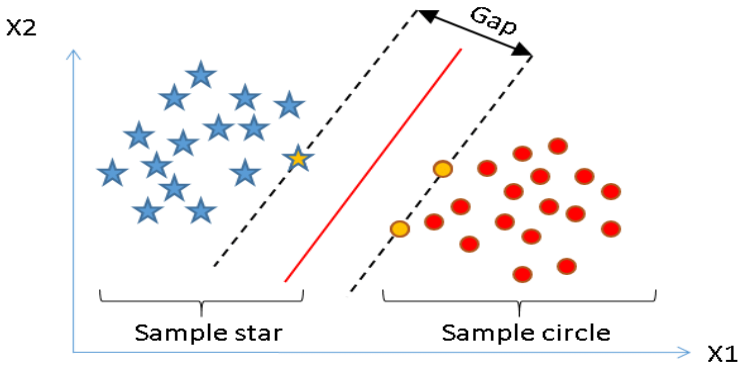


Figura 2.3: SVM Linear. Fonte: Researchgate[2]

Para problemas não linearmente separáveis é preciso aplicar um método conhecido como *Kernel Trick*, onde um plano 2D é transformado em um espaço 3D, tornando possível fazer a separação entre as duas classes. A imagem 2.4 ilustra a aplicação do *Kernel Trick* em dados não linearmente separáveis.

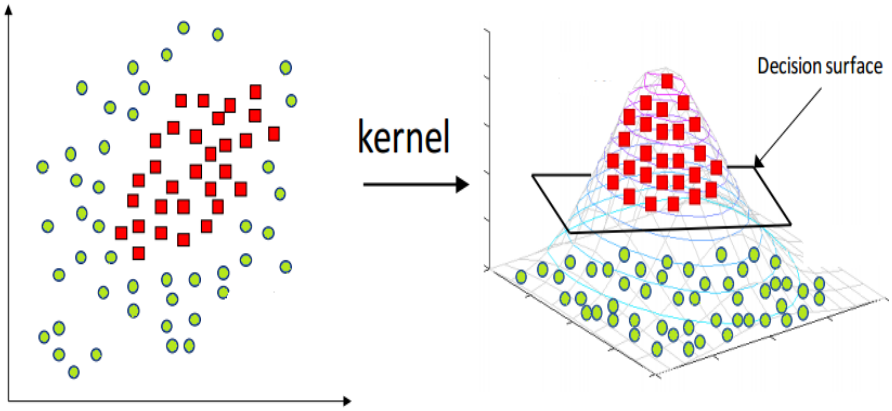


Figura 2.4: SVM Não Linear. Fonte: Medium[3]

2.2.3 K-nearest Neighbors

O KNN é um algoritmo baseado em instâncias onde um dado é classificado de acordo com a classe dos K vizinhos mais próximos, usando uma métrica de distância como a distância de Minkowski (que é uma simplificação da distância Euclidiana e de Manhattan) para medir a proximidade das intâncias mais imediatas. A imagem 2.5 ilustra o funcionamento básico

[2]Link:https://www.researchgate.net/publication/297236887_Development_of_mobile_device-based_surrogate_systems_for_connected_and_autonomous_vehicle_technologies
[3]Link:https://medium.com/@zxr.nju/what-is-the-kernel-trick-why-is-it-important-98a98db0961d

do algoritmo, onde há uma nova instância de dado a ser classificada. Quando K=3 a nova intância é classificada como sendo da classe B, mas quando K=7 ela passa a ser classificada como A. Encontrar o melhor valor para K é essencial para se conseguir bons resultados com este algoritmo.

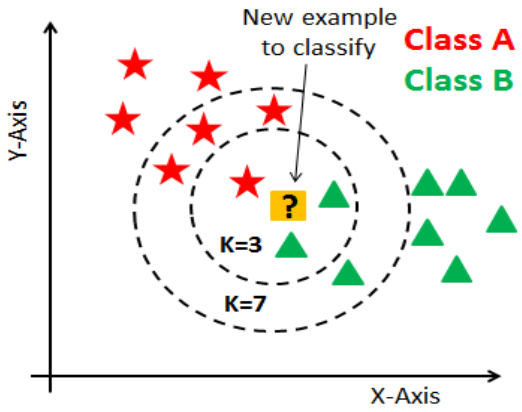


Figura 2.5: KNN. Fonte: Datacamp[4]

2.2.4 Artificial Neural Network - Multilayer Perceptron

As Redes Neurais Artificiais (Artificial Neural Networks) são dispositivos não-lineares, inspirados na funcionalidade dos neurônios biológicos, estas redes neurais são compostas de neurônios formados por modelos matemáticos, juntamente com a estrutura coletiva de uma rede, com diversos neurônios. Seu funcionamento padrão é receber dados de entrada, processar estes dados, aplicando funções de soma e ativação para então se chegar a um resultado de saída. Este processo é repetido diversas vezes até se chegar a um certo resultado.

Os componentes fundamentais de uma rede neural são:

- Entradas: as entradas são os dados de algum problema do qual a rede irá tentar aprender um padrão de comportamento;
- Pesos: as *arestas* que ligam cada neurônio das camadas da rede possuem *pesos*, estes pesos começam com valores aleatórios em uma rede não treinada, e vão sendo ajustados ao longo da aprendizagem da rede;

[4]Link:https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn

- Neurônios: são a parte essencial da rede. Funcionam como unidades de processamento de sinais e executam duas funções:
  - Função de soma: realiza o somatório dos valores das multiplicações das entradas pelos seus respectivos pesos;
  - Função de ativação: tem como entrada o resultado da função de soma e aplica sobre esse valor uma função de ativação, passando o resultado para a camada seguinte. Existem algumas funções de ativação amplamente utilizadas em redes neurais, tais como as funções tangente hiperbólica, escada e sigmoide, sendo cada função utilizada para fins específicos.
- Camadas: uma rede neural é configurada de acordo com o número de neurônios e o número de camadas. As camadas estão divididas em: *camada de entrada*, *camadas ocultas* e *camada de saída*. A primeira camada é a de entrada, cuja responsabilidade é passar os valores de entrada para a primeira camada de neurônios. Entre as camade de entrada e saída podem haver diversas camadas intermediárias, chamadas *camadas ocultas*, que contribuirão no processamento dos dados. Por fim, a camada de saída cujo número de neurônios de saída é equivalente ao número de classes do problema.

O Multilayer Perceptron (Perceptron Multicamadas) é uma rede neural que apresenta uma ou mais camadas ocultas e com um número indefinido de neurônios. Sua estrutura básica pode ser visualizada na imagem 2.6.

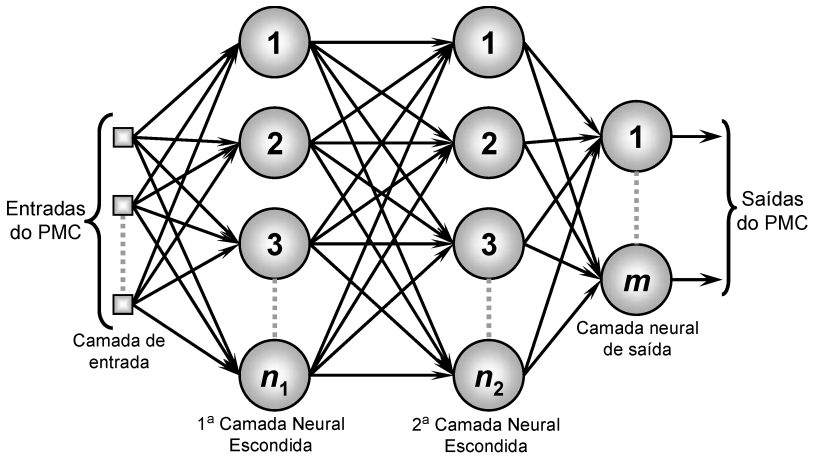


Figura 2.6: Perceptron Multicamadas (PMC). Fonte: Medium[5]

[5]Link:https://medium.com/ensina-ai/rede-neural-perceptron-multicamadas-f9de8471f1a9

3 REVISÃO BIBLIOGRÁFICA

Neste capítulo é feita uma revisão de assuntos e trabalhos relacionados ao tópico de IoT e Machine Learning. O capítulo está organizado da seguinte forma: as seções 3.1 e 3.2 focam no estudo do tráfego de redes IoT, seus tipos e funcionamento. As seções 3.3 e 3.4 falam dos atributos de rede, sendo que a seção 4 foca no uso dos atributos para a identificação de dispositivos. Por fim, a seção 3.5 explora trabalhos relacionados, dando destaque aos atributos escolhidos e algoritmos de Machine Learning utilizados, concluindo o capítulo com uma comparação dos atributos e algoritmos utilizados pelos autores.

3.1 TRÁFEGO DE REDE IOT

De acordo com (WANG et al., 2018), ao olharmos do ponto de vista da operação de redes, fazer a identificação do tipo de dispositivo (por exemplo, câmera, sensor ambiental, smart devices, etc.) pode trazer diversos benefícios. Por exemplo, dependendo da categoria do dispositivo, o administrador da rede pode configurar diferentes conjuntos de regras, restringir o uso de determinados dispositivos, ou até mesmo dar prioridade para o tráfego de rede de um dispositivo em relação a outro quando necessário.

Entretanto, fazer a classificação de dispositivos IoT baseado no tráfego de rede não é tarefa trivial, tendo em vista a sua natureza dinâmica. O tráfego gerado por um dispositivo pode variar de acordo com alguns fatores, como a hora em que a medição é feita, a interação com os usuários e a comunicação cliente-servidor do dispositivo. Além disso, cada categoria pode contar com dispositivos que apresentam funções similares, mas que possuem software/hardware diferentes, dependendo do fabricante. Portanto, um padrão de comportamento de rede por categoria de dispositivo pode ser difícil de se encontrar sem as ferramentas adequadas.

Para entender melhor o ambiente IoT como um todo, é preciso entender como os dispositivos que fazem parte dele se comunicam. Para isso é feito um estudo dos tipos mais comuns de tráfego de rede em dispositivos IoT atualmente.

3.2 TIPOS DE TRÁFEGO EM IOT

Na rede IoT, o principal tipo de tráfego gerado na comunicação entre dispositivos é do tipo M2M (Machine-to-Machine) ou MTC (Machine Type Communication), o qual diferente

da HTC (Human Type Communications), para a qual a tecnologia de rede sem fio 4G foi criada e otimizada, necessita de muito pouca intervenção humana (GARRETT et al., 2018).

Em (NIKAEIN et al., 2013) são identificados 3 padrões comuns de tráfego MTC: Periodic Update (PU), Event-driven (ED), Payload Exchange (PE).

Em PU o dispositivo transmite relatórios de status regulares para uma unidade central. PU não é em tempo real, apresenta um padrão de tempo constante e o tamanho dos dados transmitidos tem um tamanho padrão. Um exemplo de PU são os leitores de energia Smart Meters. Em ED os dados são transmitidos quando um evento é desencadeado por algum dispositivo. O padrão ED é usado principalmente para tráfego em tempo real, não havendo pradrão de tempo ou tamanho dos dados transmitidos. Um exemplo de ED pode ser um alarme para tsunamis. O tipo PE é emitido após um evento ocorrer, como o dos outros dois tipos anteriores (PU e ED) e abrange os casos em que uma quantidade maior de dados é trocada entre o dispositivo e uma entidade centralizadora. Este tráfego pode ou não ser em tempo real, dependendo do tipo de sensor e evento ocorrido. Em aplicações atuais, o mais comum é que exista uma combinação desses 3 tipos de tráfego, com os dispositivos programados para entrar em modos de comunicação diferentes de acordo com a situação que se encontram.

O tráfego de rede consiste em um grande conjunto de dados de pacotes de rede gerados durante as comunicações entre dispositivos (SANTOS, 2011), e a escolha dos atributos de tráfego a serem analisados impacta o nível de confiabilidade da classificação de comportamento dos dispositivos na rede, pois tanto a qualidade e a quantidade de atributos influenciam na precisão da classificação do comportamento do tráfego.

3.3 ATRIBUTOS DE TRÁFEGO DE REDES

O tráfego de uma rede pode ser analisado a partir de diferentes níveis de abstração, envolvendo dados de sessão, fluxo, pacotes e bytes. O que define o nível de abstração da análise do tráfego são as características ou atributos selecionados para análise.

Segundo (SANTOS, 2011), os atributos podem ser classificados em dois tipos, primitivos ou derivados. Os atributos primitivos são aqueles extraidos diretamente do cabeçalho dos pacotes de rede, como o endereço IP de origem/destino, protocolo utilizado, porta, etc. Os atributos derivados são obtidos através do processamento dos atributos primitivos, como por exemplo a duração de cada seção de captura de pacotes ou tempo médio de chegada entre os pacotes.

Nas pesquisas de (SANTOS, 2011), estudos científicos na área de tráfego de rede foram analisados, e foi feito o levantamente de alguns dos atributos com maior utilização na área de detecção de anomalias em redes. Estes atributos se encontram na tabela 3.1 e alguns deles também foram utilizados neste trabalho, para a identificação e classificação dos dados de dispositivos IoT.

Atributos	Descrição
serviço	Serviços acessados (por porta): HTTP, FTP, Telnet
duração	Duração da conexão
src_ip	Endereço IP do iniciador da conexão
dst_ip	Endereço IP do host destino
src_bytes	Número de bytes enviados pelo iniciador
dst_bytes	Número de bytes enviados pelo host destino
protocolo	TCP, UDP, ICMP
num_conn	Número de conexões abertas
tcp_flags	Flags TCP (SYN, ACK, RST, ...)

Tabela 3.1: Tabela de atributos usados por (SANTOS, 2011)

3.4 IDENTIFICAÇÃO DE DISPOSITIVOS IOT

Para fazer a identificação dos dispositivos IoT conectados a uma rede, uma das técnicas mais promissoras atualmente é a de Fingerprinting de Dispositivos.

Segundo (ANEJA; ANEJA; ISLAM, 2018), DFP (Device Fingerprinting) cria uma assinatura de rede exclusiva para a identificação de cada dispositivo, não dependendo assim de endereço de rede IP ou identidade física como endereço MAC ou número IMEI (International Mobile Equipment Identity). Em (BEZAWADA et al., 2018), DFP é descrito como a identificação de um dispositivo baseado em uma amostra de sua atividade de rede, e a coleção de todas as atividades de rede de um certo dispositivo (ou seja, o conjunto de Fingerprints do dispositivo) constitui no perfil de comportamento deste dispositivo.

As Fingerprints de um dispositivo nada mais são do que o conjunto de características de rede que podem ser examinadas no cabeçalho dos pacotes de dados enviados por estes dispositivos, como se pode ver na tabela 3.1.

Em (BEZAWADA et al., 2018) é visto que DFP pode ser usado para a classificação em vários níveis de granularidade, desde uma categoria mais geral, agrupando dispositivos com funções similares, como uma Lâmpada por exemplo, até um tipo mais específico como uma Lâmpada Monocromatica TP-Link, como mostrado na figura 3.1.

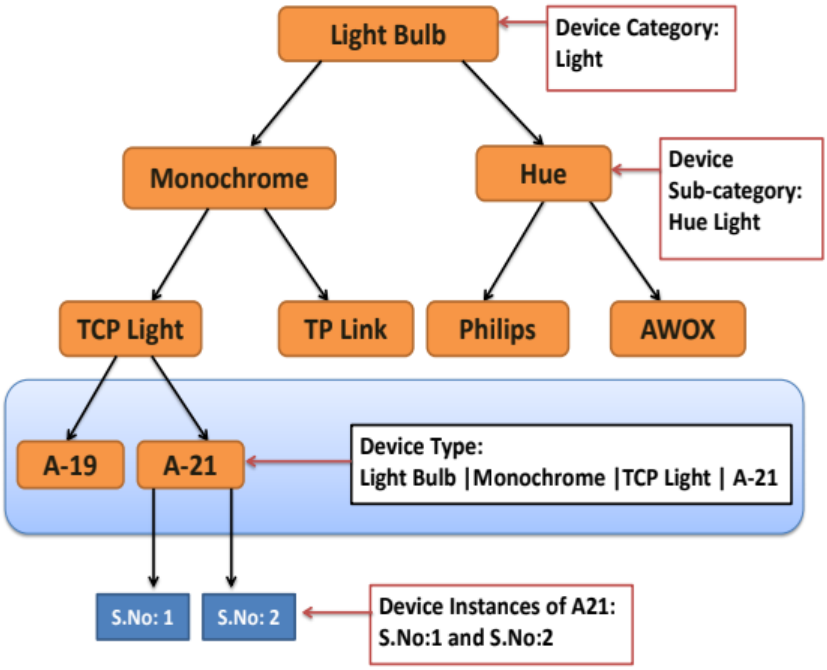


Figura 3.1: Categorização de dispositivos IoT. Fonte: (BEZAWADA et al., 2018)

3.5 ATRIBUTOS DE REDE E MACHINE LEARNING

A importância de escolher os atributos que irão constituir o DFP se dá pelo fato de que a base de dados usada para o treinamento dos classificadores de Machine Learning é construida com base nestes atributos. Com o propósito de descobrir quais atributos de tráfego de rede são considerados relevantes, a seguir é feita uma análise de estudos correlatos que abrangem este tópico.

Em (SHAIKH et al., 2018) é apresentado um modelo para classificar características de rede provenientes de dispositivos maliciosos. O trabalho tem como objetivo identificação de ataques em empresas originados de dispositivos IoT comprometidos. Para isto são utilizas informações do cabeçalho de pacotes vindos de uma base de dados da darknet.

Os classificadores de machine learning explorados foram Random Forest, Gradient Boosting, AdaBoost, Naive Bayes, dentre os quais Gradient Boost e Random Forest obtiveram os melhores resultados. As características utilizadas por (SHAIKH et al., 2018) podem ser observadas na tabela 3.2.

Característica	Descrição
TTL médio	O valor médio de ttl (time to live) para cada endereço IP de origem
Contagem total de pacotes	O número total de pacotes enviados por cada endereço IP de origem
Número de IPs de destino únicos	O número total de IPs de destino únicos direcionados por cada endereço IP de origem
Média de pacotes enviados para cada porta	O número médio de pacotes enviados para cada porta de destino por cada endereço IP de origem
Média de pacotes enviados para cada IP	O número médio de pacotes enviados para cada IP de destino por cada endereço IP de origem
Inter arrival time	A média do tempo entre chegadas dos pacotes enviados por cada IP de origem
Scan flags	O número de pacotes enviados com sinalizadores de verificação (SYN, FIN, FIN-ACK, NULL)
Pacotes Udp	O número de pacotes UDP enviados por cada endereço IP de origem
Número de portas de destino exclusivas	O número total de portas de destino exclusivas direcionadas por cada endereço IP de origem

Tabela 3.2: Tabela de atributos usados por (SHAIKH et al., 2018)

Em (MIETTINEN et al., 2017) é proposto o IOT SENTINEL, um sistema direcionado a redes pequenas como a de casas e pequenos escritórios. O sistema é capaz de identificar o tipo de dispotivo IoT introduzido na rede e impor medidas para mitigar possíveis falhas na segurança de dispositivos considerados vulneráveis. O IOT SENTINEL controla o tráfego de rede acessado pelos dispositivos reconhecidos como vulneráveis para evitar problemas a outros dispositivos conectados na rede. Para fazer a identificação de dispositivos IoT foi feita a coleta do tráfego de rede de 27 dispositivos distintos. Estes dados foram utilizados pelos autores para o treinamento de classificadores Random Forest binários para cada tipo de dispositivo, sendo os dados compostos pelas caracteristicas apresentadas na tabela 3.3.

Tipo	Características
Protocolo de camada de enlace	ARP / LLC
Protocolo de camada de rede	IP / ICMP / ICMPv6 / EAPoL
Protocolo da camada de transporte	TCP / UDP
Protocolo da camada de aplicação	HTTP / HTTPS / DHCP / BOOTP /SSDP / DNS / MDNS / NTP
Opções de IP	Padding / RouterAlert
Conteúdo do pacote	Tamanho / Dados brutos
Endereço de IP	Contador de IP de destino
Classe de porta	Origem / Destino

Tabela 3.3: Tabela de atributos usados por (MIETTINEN et al., 2017)

A proposta de (WANG et al., 2018) é um método de classificação automática para dispositivos não previamente registrados na rede.

Sua abordagem leva em consideração características da perspectiva de volume de tráfego, comprimento de pacotes, protocolos de rede e direção do tráfego, além da característica temporal do tráfego de rede.

Estas características são usadas para treinar uma LSTM-CNN (Long Short-Term Memory - Convolutional Neural Network) para fazer a classificação de dispositivos IoT desconhecidos em categorias de acordo com suas funções, sendo LSTM uma variação da RNN (Recurrent Neural Networks) projetada especialmente para o processamento de dados sequenciais. As características de tráfego utilizadas neste artigo podem ser observadas na tabela 3.4.

Atributos	Descrição
t	Tempo de chegada/envio do pacote
length	Tamanho em bytes do pacote
protocol	Protocolo utilizado
eth.src	Origem do pacote
eth.dst	Destino do pacote

Tabela 3.4: Tabela de atributos usados por (WANG et al., 2018)

Em (ANEJA; ANEJA; ISLAM, 2018) o objetivo foi identificar dispositivos IoT baseando-se em apenas 1 atributo, o Inter Arrival Time (IAT), ou seja, o intervalo de tempo entre dois pacotes consecutivos recebidos. Para isso eles geram imagens de gráficos a partir dos IAT dos pacotes registrados, os quais servem de entrada para uma Convolution Neural Network (CNN). No fim, a taxa de acerto alcançada foi de 86.7%.

No artigo de (SHAHID et al., 2019) são apresentadas técnicas baseadas em Machine Learning para Monitoramento de redes IoT. Eles analisam tanto o tráfego de rede IoT comum quanto tráfego de rede malicioso, e utilizam características como:

- O tamanho dos primeiros N pacotes enviados;
- O tamanho dos primeiros N pacotes recebidos;
- Os inter-arrival times de N-1 pacotes entre os primeiros N pacotes enviados;
- Os inter-arrival times de N-1 pacotes entre os primeiros N pacotes recebidos.

Em relação aos classificadores, são testados diversos algoritmos, sendo eles: Decision

Tree, Random Forest, Naive Bayes, KNN, SVM e ANN, dentre os quais o Random Forest apresentou os melhores resultados.

Abaixo, nas tabelas 3.5 e 3.6, podem ser vistos os atributos e algoritmos utilizados respectivamente por cada autor, sendo possível analisar os que aparecem em mais de um trabalho.

Atributo	Autor
Protocolo	(SANTOS, 2011), (MIETTINEN et al., 2017), (WANG et al., 2018)
Duração	(SANTOS, 2011)
IP (src/dst)	(SANTOS, 2011), (MIETTINEN et al., 2017)
Tamanho do pacote	(SANTOS, 2011), (MIETTINEN et al., 2017), (WANG et al., 2018), (SHAHID et al., 2019)
n° de conexões	(SANTOS, 2011)
flags	(SANTOS, 2011), (SHAIKH et al., 2018)
Time to Live	(SHAIKH et al., 2018)
Total de pacotes	(SHAIKH et al., 2018)
nº de IPs únicos	(SHAIKH et al., 2018)
Média de pacotes por porta	(SHAIKH et al., 2018)
Média de pacotes por IP	(SHAIKH et al., 2018)
Inter Arrival Time	(SHAIKH et al., 2018), (ANEJA; ANEJA; ISLAM, 2018), (SHAHID et al., 2019)
nº portas únicas	(SHAIKH et al., 2018)
nº de pacotes udp	(SHAIKH et al., 2018)
Opções IP	(MIETTINEN et al., 2017)
Porta	(MIETTINEN et al., 2017)
Tempo de chegada	(WANG et al., 2018)
Endereço ethernet	(WANG et al., 2018)

Tabela 3.5: Tabela de atributos usados nos trabalhos relacionados.

Algoritmos	Autor
Decision Tree	(SHAHID et al., 2019)
Random Forest	(SHAIKH et al., 2018), (MIETTINEN et al., 2017), (SHAHID et al., 2019)
Gradient Boost	(SHAIKH et al., 2018)
Ada Boost	(SHAIKH et al., 2018)
Naive Bayes	(SHAIKH et al., 2018), (SHAHID et al., 2019)
LSTM-CNN	(WANG et al., 2018)
CNN	(ANEJA; ANEJA; ISLAM, 2018)
SVM	(SHAHID et al., 2019)
KNN	(SHAHID et al., 2019)
ANN	(SHAHID et al., 2019)

Tabela 3.6: Tabela de algoritmos usados nos trabalhos relacionados.

No trabalho aqui apresentado o foco não é apenas em um aspecto de classificação, mas abrange tanto a identificação de dados entre diferentes tipos de dispositivos quanto a diferenciação entre tráfego de rede IoT padrão e malicioso.

4 CLASSIFICAÇÃO DE DISPOSITIVOS IOT

Neste capítulo é apresentada a metodologia utilizada na execução da classificação de dispositivos IoT. Na seção 4.1 é visto o tratamento dos dados usados nos algoritmos de machine learning, na seção 4.2 são apresentados os detalhes de configuração dos algoritmos utilizados e na seção 4.3 são vistas as métricas de avaliação dos classificadores implementados.

4.1 TRATAMENTO DOS DADOS

Para se conseguir dados de dispositivos IoT o ideal seria montar um ambiente com diversos dispositivos e capturar o tráfego gerado na rede . Como isso não é viável para este trabalho, a opção escolhida foi usar uma base de dados préviamente construida. Devido a área de estudos de IoT ser algo relativamente novo, o número de bases de dados publicamente disponíveis para uso ainda é escasso. Então, para este trabalho foi escolhida a base montada por (MIETTINEN et al., 2017) que se encontra disponível online[6]. Esta base representa o tráfego emitido durante a configuração de 27 dispositivos IoT em uma *Smart Home* e se encontra dividida em arquivos .pcap de acordo com o tipo de dispositivo. A partir destes arquivos foram extraidos um total de 10 características para servirem de base para o DFP, as quais podem ser observadas na tabela 4.1.

Algumas destas características foram escolhidas com base nos trabalhos relacionados vistos no capítulo anterior e outras foram escolhidas por serem relevantes no contexto deste trabalho, por exemplo, o atributo *TCP Flags* pode ajudar a indicar se um pacote faz parte de um ataque *DDoS* do tipo *SYN Flood*, e o atributo *Inter-Arrival Time* pode indicar se um pacote esta fora do padrão em relação ao tempo de chegada entre pacotes. Além disso, apenas atributos primitivos foram escolhidos, já que o trabalho lida com classificação de pacotes individuais, e não do fluxo de pacotes em conjunto.

[6] A base de dados pode ser acessada através do seguinte link: https://www.kaggle.com/drwardog/iot-device-captures

Atributos	Descrição
Time	Tempo de chegada do pacote
Protocol	Protocolo utilizado
Length	Tamanho do pacote em bytes
Src Port	Porta de origem
Dst Port	Porta de destino
Inter-Arrival Time	O tempo entre a chegada de um pacote e seu anterior
Ethernet Type	O tipo Ethernet (Arp, ipv4, 802.1X Auth, ...)
TTL	Time to live
IP Protocol	TCP, UDP, ...
TCP Flags	SYN, ACK, FIN, etc

Tabela 4.1: Atributos para DFP usados em todas as bases de dados

Com as características escolhidas, a extração dos dados é feita, transformados os arquivos .pcap em .csv para uso posterior nos classificadores de machine learning.

4.1.1 Criação das bases de classificação

Para criar as bases de dados de treinamento e teste para cada classificador a seguinte metodologia foi adotada:

Primeiramente, os arquivos .csv contendo os dados de tráfego de rede de cada dispositivo foram concatenados em um arquivo único. Em seguida, para cada tipo de dispositivo existente fora criada uma base separada, contendo todos os dados dos pacotes de rede deste dispositivo e mais X dados de pacotes aleatórios dos outros 26 dispositivos, onde X equivale ao número total de pacotes do dispositivo a ser classificado. Sendo assim, cada base é formada por 50% de dados do dispositivo a ser classificado e 50% com amostras de dados dos outros dispositivos.

4.1.2 Tratamento dos atributos

Com as bases dos classificadores prontas, o passo seguinte foi adaptar os atributos para que pudessem ser de fato utilizados nos classificadores.

Grande parte dos dados apresentavam valores categóricos (strings) na base de dados original, por exemplo, "Protocol"= "EAPOL"ou então "Ethernet Type"= "IPv4", então foi necessário fazer a conversão destes atributos para valores numéricos, que são aceitos pelos algoritmos de ML. Além disso, nestes atributos também é aplicado o método *One-hot Encoding*, onde para cada valor distinto de uma caracteristica é criada uma nova coluna, sendo assim criadas várias colunas novas, o que aumenta o tamanho da base de dados de acordo com a variedade das

característica. Estas colunas apresentam valores binários (0 ou 1), onde 0 significa que a característica não esta presente no pacote, e 1 significa que esta presente. Um exemplo de *One-hot Encoding* pode ser visto na figura 4.1. Este método não foi aplicado a todos os atributos, apenas àqueles que apresentavam valores categóricos, com os atributos númericos permanecendo iguais, e seu uso foi devido à melhora dos resultados apresentados pelos classficadores com sua aplicação.

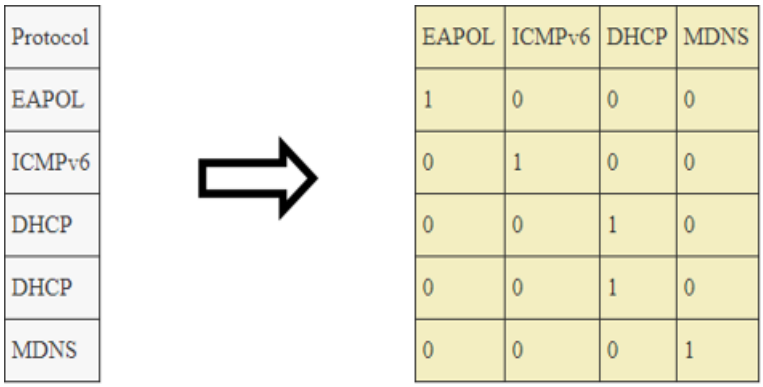


Figura 4.1: One-hot Encoding exemplo. Fonte: O autor.

Outro tratamento feito é a transformação dos valores dos atributos de "Src Port"e "Dst Port". Como a variação do número da porta e seus valores é muito grande, é feito a transformação apresentada em (MIETTINEN et al., 2017), que divide as portas em 4 classes de rede distintas. A transformação pode ser observada na tabela 4.2.

Classe de porta de rede	Número da porta	Valor atribuído
Nenhuma porta	Sem número	0
Porta conhecida	0 - 1023	1
Porta registrada	1024 - 49151	2
Porta dinâmica	49152 - 65535	3

Tabela 4.2: Conversão dos valores de porta de rede

4.2 CONFIGURAÇÃO DOS ALGORITMOS DE MACHINE LEARNING

Tendo feito o tratamento das bases de dados, o próximo passo é fazer o treinamento e teste dos classificadores.

Para a etapa de treinamento e teste foi utilizado o método *Stratified K-Folds cross-validation*, em que é feita a divisão da base de dados K vezes entre bases de teste e base de

treinamento, em K partes separadas do mesmo tamanho, sendo que em cada uma das K divisões, K-1 bases são usadas para treinamento e 1 base para teste. A figura 4.2 ilustra o funcionamento das divisões da base para K=5. Este método faz uma boa distribuição entre as classes de dados existentes na base de dados e também aumenta a confiabilidade dos resultados obtidos pelos classificadores. Foram feitas configurações de testes com 4, 6 e 10 folds, onde o valor K=10 (Stratified 10-Folds cross-validation) apresentou os melhores resultados, sendo assim o valor definitivo escolhido para divisão da base. Após executar 10 vezes os testes com cada algoritmo classificador para cada dispositivo, é feita uma média dos resultados de *acurácia*, *precisão* e *revocação* obtidos por dispositivo. A biblioteca utilizada foi a *sklearn* da linguagem *Python*, e a função *sklearn.model_selection.StratifiedKFold*.

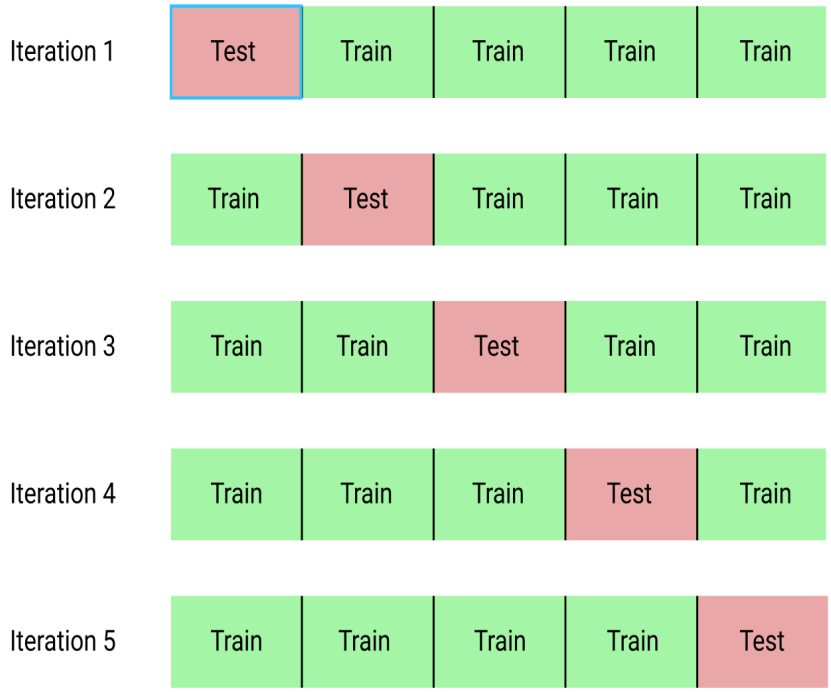


Figura 4.2: Stratified K-Folds cross-validation. Fonte: Towardsdatascience[7]

Um total de 4 algoritmos foram escolhidos para serem classificadores, sendo eles: Random Forest, Multilayer Perceptron, K-nearest Neighbor e Support Vector Machine. Esta escolha se deu principalmente devido ao desempenho dos mesmos em trabalhos relacionados, mas também pelo nível de familiaridade do autor com os mesmos. Todos eles foram feitos e testados

[7]Link:https://towardsdatascience.com/cross-validation-explained-evaluating-estimator-performance-e51e5430ff85

utilizando a linguagem de programação *Python*, fazendo uso da biblioteca *sklearn*. A seguir, na tabela 4.3 estão as configurações individuais para cada algoritmo.

Classificadores	**Configurações utilizadas nos classificadores**	
Random Forest	Função:	Parâmetros:
	from sklearn.ensemble import `RandomForestClassifier`	*n_estimators* = 40 *criterion = "entropy"*
Multilayer Perceptron	Função:	Parâmetros:
	from sklearn.neural_network import `MLPClassifier`	*max_iter = 1000* *tol = 0.000010* *solver = "adam"* *hidden_layer_sizes = (100)* *activation = "relu"* *batch_size = 200* *learning_rate_init = 0.001*
K-nearest Neighbors	Função:	Parâmetros:
	from sklearn.neighbors import `KNeighborsClassifier`	*n_neighbors = 5* *metric = "minkowski"* *p = 2*
Support Vector Machine	Função:	Parâmetros:
	from sklearn.svm import `SVC`	*kernel = "rbf"* *C = 2.0*

Tabela 4.3: Configurações dos classificadores em python.

Estes parâmetros foram estabelecidos após vários testes para se tentar alcançar os melhores resultados.

4.3 AVALIAÇÃO DOS ALGORITMOS

As métricas utilizadas para avaliar a eficiência dos algoritmos são a Acurácia (Accuracy), Precisão (Precision) e Revocação (Recall), cujas definições descritas na literatura de machine learning (SHAIKH et al., 2018) podem ser entendidas como:

Accuracy é a proporção das classes rotuladas corretamente para todo o conjunto de classes.

Precision dentre todas as classificações de classe Positivo que o modelo fez, quantas estão corretas.

Recall dentre todas as situações de classe Positivo como valor esperado, quantas estão corretas.

Estas métricas foram escolhidas por serem as mais comumente utilizadas para se fazer

a validação da eficiência de classificadores em Machine Learning, como pode ser visto em (SANTOS; CALLADO, 2017), (NGUYEN; ARMITAGE, 2008) e (WANG, 2013).

As fórmulas de cada métrica podem ser observadas abaixo:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

Figura 4.3: Métricas tradicionais usadas em Machine Learning: Acurácia, Precisão e Revocação

Para calcular estas métricas são necessários os valores da matriz de confusão gerada após a classificação das bases de dados. Estes valores são: Verdadeiros Negativos (True Negative - TN), Falsos Negativos (False Negative - FN), Verdadeiros Positivos (True Positive - TP) e Falsos Positivos (False Positive - FP).

Tomando como exemplo o classificador de um dos dispositivos cujos pacotes de dados são utilizados no trabalho, a *balança smart* Aria, TN representa a quantidade de dados que não são do tipo Aria e foram classificados como não sendo do tipo Aria, FN representa a quantidade de dados que são do tipo Aria e foram classificados como não sendo, TP representa os dados que são do tipo Aria e foram classificados como sendo, e FP representa os dados que não são do tipo Aria e foram classificados como sendo.

O modelo da matriz de confusão pode ser observado na tabela 4.4.

	Valor previsto		
Valor real		Negative	Positive
	Negative	True Negative (TN)	False Positive (FP)
	Positive	False Negative (FN)	True Positive (TP)

Tabela 4.4: Matriz de Confusão

5 RESULTADOS DE CLASSIFICAÇÃO DOS DISPOSITIVOS

Neste capítulo é apresentado, na seção 5.1, as médias de acurácia, precisão e revocação encontradas por cada classificador e na seção 5.2 os testes estatísticos realizados e resultados obtidos.

5.1 RESULTADOS DE CLASSIFICAÇÃO

Os resultados obtidos na classificação e identificação de dispositivos podem ser observados nas tabelas 5.1, 5.2 e 5.3 que mostram os valores de acurácia, precisão e revocação alcançados respectivamente por cada um dos classificadores. Nas figuras 5.1, 5.2 e 5.3 estão as representações gráficas dos resultados.

A tabela 5.1 apresenta os valores de acurácia de cada classificador. Ao observarmos os resultados, verificamos que o algoritmo de *RF* obteve uma média de acurácia de 95.9% entre todos os dispositivos, sendo o algoritmo que obteve os melhores resultados nesta métrica. Os algoritmos *MLP*, *SVM* e *KNN* obtiveram médias de acurácia de 93.36%, 92.28% e 92.42% respectivamente.

A tabela 5.2 apresenta os resultados de precisão e a tabela 5.3 mostra os resultados de revocação obtidos pelos classificadores. Em ambas as métricas o classificador Random Forest apresentou os melhores resultados, assim como em acurácia, apresentando médias de 97.37% e 94.28% de precisão e revocação respectivamente. Os classificadores *MLP*, *SVM* e *KNN* obtiveram médias de precisão de 95.22%, 94.14% e 95.35% e médias de revocação de 91.39%, 90.11% e 89.14% respectivamente.

	Acurácia			
Dispositivo	Random Forest	KNN	SVM	Multilayer Perceptron
Aria	0.98857	0.97403	0.98286	0.98546
D-LinkCam	0.96454	0.91076	0.88801	0.90975
D-LinkDayCam	0.99636	0.97814	0.98421	0.99393
D-LinkDoorSensor	0.95364	0.90306	0.90134	0.91043
D-LinkHomeHub	0.94395	0.90140	0.86626	0.88692
D-LinkSensor	0.87991	0.85072	0.82354	0.81688
D-LinkSiren	0.88513	0.86502	0.83268	0.82968
D-LinkSwitch	0.89988	0.87000	0.82965	0.83475
D-LinkWaterSensor	0.88767	0.85998	0.82997	0.84080
EdimaxCam	0.99386	0.97490	0.98550	0.99609
EdimaxPlug1101W	0.95693	0.89470	0.90034	0.93666
EdimaxPlug2101W	0.95987	0.88715	0.89922	0.92607
EdnetCam	0.99890	0.94591	0.96578	0.99890
EdnetGateway	0.98561	0.96526	0.97018	0.97439
HomeMaticPlug	0.99676	0.96807	0.97409	0.99306
HueBridge	0.89751	0.85195	0.90398	0.84539
HueSwitch	0.94667	0.91857	0.93513	0.93048
iKettle2	0.98899	0.96275	0.98029	0.98024
Lightify	0.99825	0.98033	0.98127	0.99710
MAXGateway	0.98851	0.97277	0.97191	0.97830
SmarterCoffee	0.97309	0.92968	0.95030	0.96267
TP-LinkPlugHS100	0.96931	0.91494	0.92863	0.96007
TP-LinkPlugHS110	0.97844	0.94753	0.95485	0.96746
WeMoInsightSwitch	0.94899	0.91588	0.89521	0.91558
WeMoLink	0.96831	0.93253	0.90889	0.92885
WeMoSwitch	0.95045	0.91823	0.90071	0.92244
Withings	0.98987	0.95912	0.96997	0.98553
Médias	0.95900	0.92423	0.92282	0.93364

Tabela 5.1: Resultados de acurácia dos classificadores.

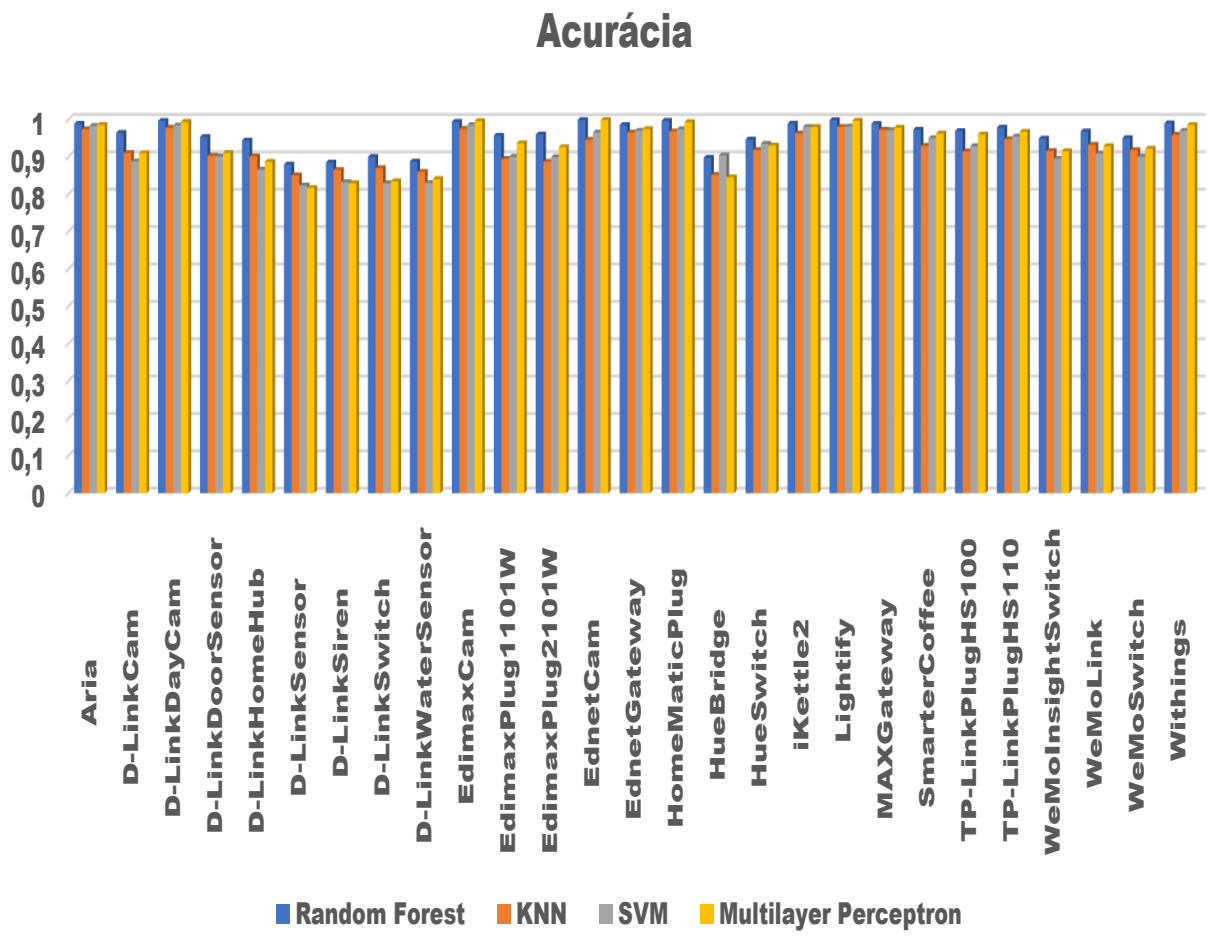


Figura 5.1: Gráfico de acurácia dos classificadores. Fonte: O autor.

	Precisão			
Dispositivo	Random Forest	KNN	SVM	Multilayer Perceptron
Aria	0.99595	0.99149	0.99694	0.99269
D-LinkCam	0.97050	0.93591	0.87554	0.91063
D-LinkDayCam	0.99913	0.99915	0.99919	0.99615
D-LinkDoorSensor	0.96332	0.92213	0.90421	0.90611
D-LinkHomeHub	0.95214	0.91301	0.85306	0.87313
D-LinkSensor	0.91304	0.88936	0.86777	0.84184
D-LinkSiren	0.91973	0.91042	0.89127	0.84817
D-LinkSwitch	0.92501	0.90624	0.84436	0.87919
D-LinkWaterSensor	0.92276	0.90124	0.87368	0.88573
EdimaxCam	1.00000	0.99778	0.99767	1.00000
EdimaxPlug1101W	0.97423	0.93846	0.92836	0.95522
EdimaxPlug2101W	0.98311	0.93045	0.92089	0.94384
EdnetCam	1.00000	0.99297	0.98555	1.00000
EdnetGateway	0.99496	0.98534	0.98480	0.99149
HomeMaticPlug	0.99904	0.98502	0.98085	0.99708
HueBridge	0.91351	0.88812	0.92692	0.89991
HueSwitch	0.97066	0.95421	0.95714	0.98073
iKettle2	1.00000	0.99583	0.98669	1.00000
Lightify	0.99919	0.99484	0.99692	0.99918
MAXGateway	0.99135	0.97528	0.96996	0.98344
SmarterCoffee	0.99145	0.97584	0.96404	0.98277
TP-LinkPlugHS100	0.98838	0.95092	0.94614	0.98423
TP-LinkPlugHS110	0.99340	0.99061	0.97764	0.98629
WeMoInsightSwitch	0.96927	0.94313	0.93582	0.95670
WeMoLink	0.98586	0.95625	0.94271	0.95399
WeMoSwitch	0.97739	0.93806	0.92840	0.96329
Withings	0.99557	0.98143	0.98180	0.99704
Médias	0.97370	0.95353	0.94142	0.95224

Tabela 5.2: Resultados de precisão dos classificadores.

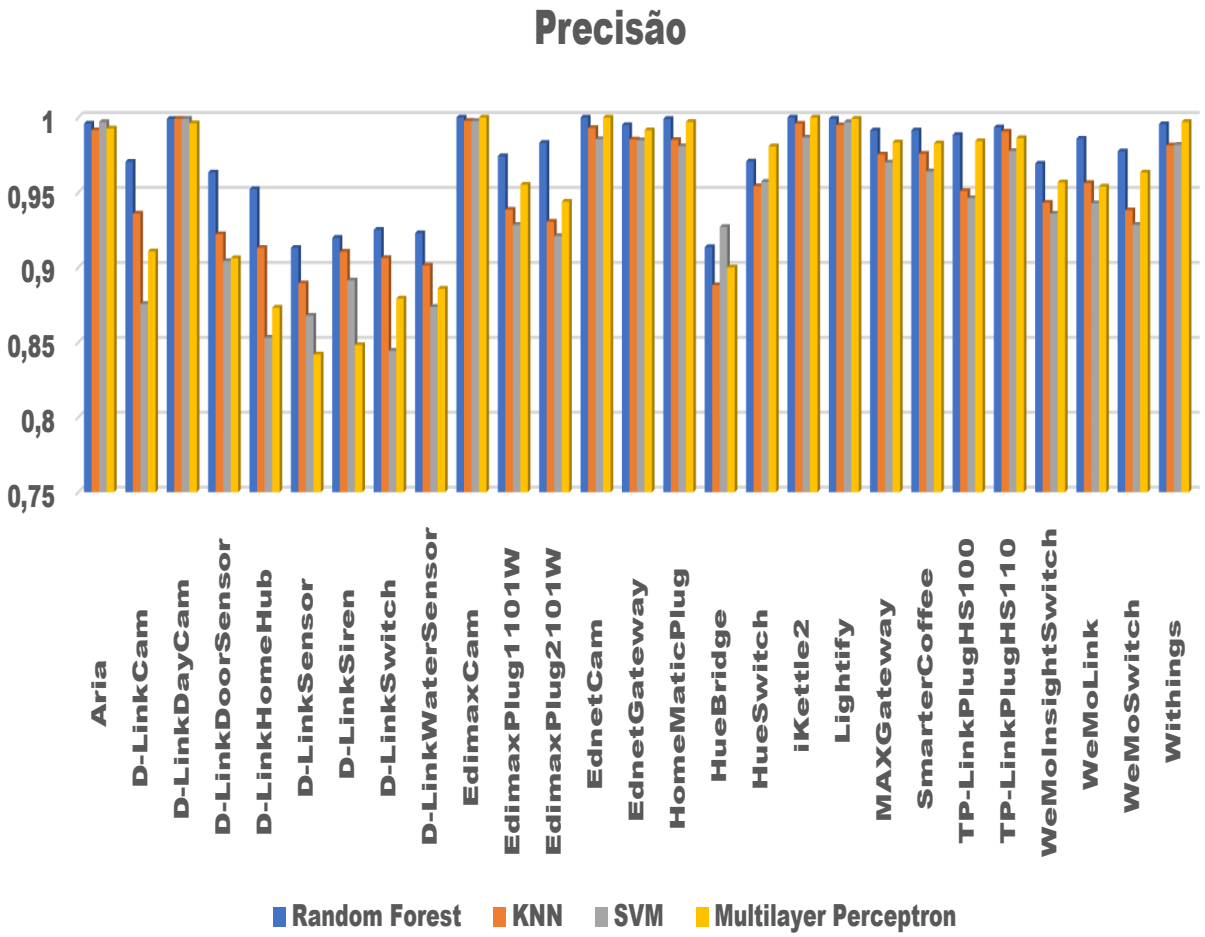


Figura 5.2: Gráfico de precisão dos classificadores. Fonte: O autor.

	Revocação			
Dispositivo	Random Forest	KNN	SVM	Multilayer Perceptron
Aria	0.98129	0.95600	0.96848	0.97826
D-LinkCam	0.95820	0.88181	0.90461	0.91191
D-LinkDayCam	0.99369	0.95711	0.96908	0.99212
D-LinkDoorSensor	0.94333	0.88057	0.89800	0.91599
D-LinkHomeHub	0.93486	0.88743	0.88502	0.90658
D-LinkSensor	0.83974	0.80106	0.76348	0.78667
D-LinkSiren	0.84377	0.80970	0.75781	0.80838
D-LinkSwitch	0.87048	0.82550	0.80841	0.78110
D-LinkWaterSensor	0.84619	0.80880	0.77158	0.78616
EdimaxCam	0.98774	0.95233	0.97386	0.99222
EdimaxPlug1101W	0.93895	0.84545	0.86794	0.91751
EdimaxPlug2101W	0.93571	0.83640	0.87231	0.90654
EdnetCam	0.99778	0.89783	0.94526	0.99778
EdnetGateway	0.97644	0.94456	0.95473	0.95701
HomeMaticPlug	0.99460	0.95041	0.96697	0.98930
HueBridge	0.87803	0.80531	0.87634	0.78670
HueSwitch	0.92120	0.87929	0.91070	0.87831
iKettle2	0.97797	0.92912	0.97301	0.95940
Lightify	0.99728	0.96561	0.96551	0.99498
MAXGateway	0.98579	0.97025	0.97347	0.97312
SmarterCoffee	0.95469	0.88053	0.93831	0.94338
TP-LinkPlugHS100	0.95006	0.87416	0.90870	0.93559
TP-LinkPlugHS110	0.96312	0.90464	0.93139	0.94900
WeMoInsightSwitch	0.92742	0.88527	0.84865	0.87127
WeMoLink	0.95017	0.90645	0.87077	0.90465
WeMoSwitch	0.92204	0.89540	0.86799	0.87842
Withings	0.98382	0.93585	0.95710	0.97365
Médias	0.94280	0.89140	0.90112	0.91390

Tabela 5.3: Resultados de revocação dos classificadores.

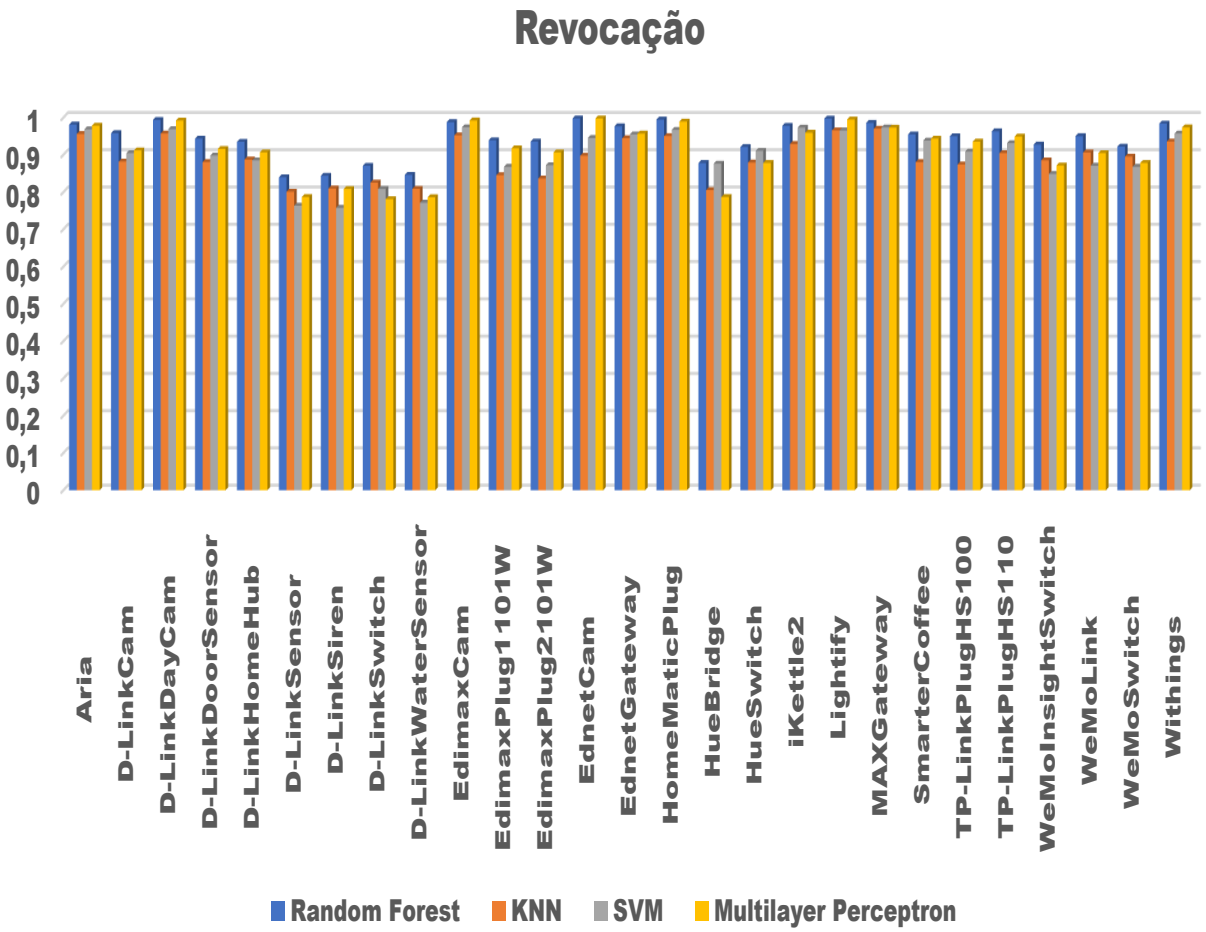


Figura 5.3: Gráfico de revocação dos classificadores. Fonte: O autor.

Com estes resultados é possível observar que o algoritmo Random Forest teve os classificadores com as melhores métricas para a grande maioria dos dispositivos. Os resultados obtidos podem ter sido bastante influenciados pelo tamanho da base de dados, talvez com bases maiores e com mais dados de treinamento o Multilayer Perceptron se saísse melhor, mas para este caso específico o Random Forest se sobressai dos demais algoritmos.

5.2 TESTES ESTATÍSTICOS

Para determinar se existe diferença estatisticamente significante entre as médias dos resultados foi feito o teste de Friedman e Nemenyi (GRANATYR, 2017) nos resultados de acurácia dos classificadores. A métrica de acurácia foi escolhida neste caso por representar o valor total de acertos de todos os dados da base, sendo assim a métrica mais importante neste estudo. Primeiramente foi feito o ranqueamento dos resultados de acurácia dos classificadores,

como pode ser observado na tabela 5.4. O ranqueamento se dá por dispositivo, ficando assim um ranqueamento por linha. Na linha do dispositivo EdnetCam os classificadores Random Forest e Multilayer Perceptron ficaram empatados, por isso o resultado não é um número inteiro.

	Posição (Ranking)			
Dispositivo	Random Forest	KNN	SVM	Multilayer Perceptron
Aria	1	4	3	2
D-LinkCam	1	2	4	3
D-LinkDayCam	1	4	3	2
D-LinkDoorSensor	1	3	4	2
D-LinkHomeHub	1	2	4	3
D-LinkSensor	1	2	3	4
D-LinkSiren	1	2	3	4
D-LinkSwitch	1	2	4	3
D-LinkWaterSensor	1	2	4	3
EdimaxCam	2	4	3	1
EdimaxPlug1101W	1	4	3	2
EdimaxPlug2101W	1	4	3	2
EdnetCam	1.5	4	3	1.5
EdnetGateway	1	4	3	2
HomeMaticPlug	1	4	3	2
HueBridge	2	3	1	4
HueSwitch	1	4	2	3
iKettle2	1	4	2	3
Lightify	1	4	3	2
MAXGateway	1	3	4	2
SmarterCoffee	1	4	3	2
TP-LinkPlugHS100	1	4	3	2
TP-LinkPlugHS110	1	4	3	2
WeMoInsightSwitch	1	2	4	3
WeMoLink	1	2	4	3
WeMoSwitch	1	3	4	2
Withings	1	4	3	2

Tabela 5.4: Ranking dos resultados de acurácia dos classificadores.

Os valores da tabela 5.4 foram usados como entrada de uma função que executa o teste de Friedman e Nemenyi. Para fazer este teste foi utilizada a linguagem de programação R e a função *nemeny* da biblioteca *tsutils*. Executando então a função: `tsutils::nemenyi(M, conf.level=0.95,plottype="vline")`, sendo M a matriz formada com os dados da tabela 5.4, obteve-se o resultado apresentado na figura 5.4.

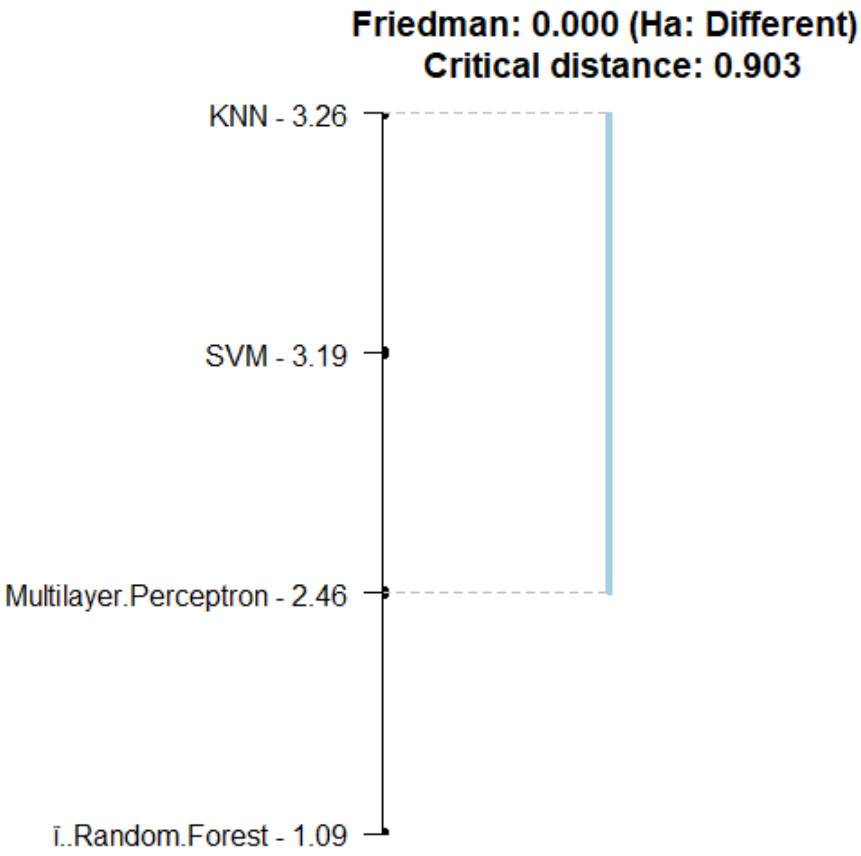


Figura 5.4: Resultado do teste de Friedman-Nemenyi. Fonte: O autor.

A distância crítica (*CD - Critical Distance*) mostrada na figura 5.4 é de 0.903, ou seja, as distâncias entre os resultados dos algoritmos devem ser maiores do que este valor para haver diferença estatística significativa (GRANATYR, 2017). A barra azul na figura 5.4 indica que não há diferença significativa do Multilayer Perceptron até o KNN, ou seja, não há diferença entre usar qualquer um destes 3 algoritmos neste caso. Mas a diferença entre Random Forest e Multilayer Perceptron (2.46 - 1.09 = 1.37) é maior do que a CD (1.37 > 0.903), ou seja, estatisticamente neste caso é melhor usar o Random Forest do que usar o Multilayer Perceptron ou qualquer um dos outros dois algoritmos. Por apresentar os melhores resultados, o algoritmo de Random Forest foi escolhido como classificador na parte de verificação de dados anômalos que é apresentada no próximo capítulo.

6 ANOMALIAS EM REDES IOT

Fazer a identificação do tipo de dispositivo é apenas o começo quando se está interessado em assegurar o funcionamento e segurança de uma rede IoT. Para garantir que nenhum dispositivo apresenta risco a integridade da rede é preciso saber quando os mesmos estão apresentando comportamentos fora do padrão.

Ataques como Mirai Botnet (MA et al., 2017) evidenciam a necessidade de se estabelecer uma maior segurança em dispositivos IoT. O ataque Mirai teve como alvo diversos dispositivos IoT, mas a escolha dos alvos não foi por acaso, levando em consideração os fabricantes dos dispositivos e seus consumidores. Muitos usuários mantêm o nome de usuário e senha padrão de fabricação dos dispositivos, tornando-os vulneráveis a ataques. O Mirai após identificar possíveis vítimas, infecta os dispositivos que apresentam estas configurações padrão através de login por força bruta, no qual ele estabelece uma conexão Telnet usando pares de nome de usuário e senha de uma lista pré configurada de credenciais. Uma possível fraqueza do Mirai é o fato de que para tentar esconder sua presença no dispositivo, além de mascarar seu nome de processo, ele deleta os arquivos binários baixados, fazendo com que a infecção não persista após uma reinicialização de sistema. Por isso é tão importante conhecer o funcionamento de rede padrão de um dispositivo IoT, para identificar atividades atípicas que podem ser possíveis infecções por malware e combatê-las como for possível.

Tendo conseguido os melhores resultados na identificação de dispositivos no capítulo anterior com o classificador Random Forest, este algoritmo foi escolhido para identificar dados de rede anômalos.

O restante do capítulo está organizado da seguinte forma: a seção 6.1 apresenta a base de dados utilizada, que contém dados anômalos, na seção 6.2 é feita a classificação separada para cada dispositivo e em seguida, na seção 6.3 é feita a classificação por categoria de dispositivos, por fim, na seção 6.4 é feita uma análise dos resultados finais.

6.1 BASE DE DADOS ANÔMALOS

Para a criação de uma base com dados anômalos foram utilizados os dados da base Bot-IoT (KORONIOTIS et al., 2018), que pode ser encontrada online[8]. A base incorpora tráfego

[8] A base de dados pode ser acessada através do seguinte link: https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/bot_iot.php

botnet legítimo e simulado, e contém dados de diversos tipos de ataques conhecidos de um ambiente IoT. Para este trabalho foram escolhidas as bases com dados de ataques *DDoS* do tipo *http flood*, *tcp flood* e *udp flood*. Esta escolha foi feita principalmente devido aos atributos de *fingerprinting* utilizados, com os quais é possível fazer a distinção entre pacotes de dados comuns e maliciosos. A base de dados anômalos é composta por pacotes de cada tipo de ataque *DDoS* (*http flood*, *tcp flood* e *udp flood*), onde cada tipo de ataque representa 1/3 da base de dados, tendo no total o mesmo número de pacotes que a base de dispositivos IoT de (MIETTINEN et al., 2017) usada para a identificação de dispositivos.

6.2 IDENTIFICAÇÃO DE ANOMALIAS POR DISPOSITIVO

Para fazer a classificação dos pacotes de rede em dados normais ou anômalos foi utilizado o classificador Random Forest com as mesmas configurações apresentadas na seção 3.2. As bases de dados para cada classificador são compostas 50% por pacotes do dispositivo a ser testado, e 50% por pacotes de ataques *DDoS* provenientes da base de dados anômalos montada a partir da base Bot-IoT disponibilizada por (KORONIOTIS et al., 2018). Para a divisão entre base de treinamento e teste foi usado novamente o método *Stratified 10-Folds cross-validation* e as métricas de avaliação *acurácia*, *precisão* e *revocação*. Os resultados podem ser observados na tabela 6.1 e em sua representação gráfica na figura 6.1. Os resultados são bastante satisfatórios, com valores de acurácia, precisão e revocação variando entre 99% a 100%.

6.3 IDENTIFICAÇÃO DE ANOMALIAS POR CATEGORIA DE DISPOSITIVO

Com bons resultados na diferenciação entre dados IoT comuns e anômalos apresentados pelos classificadores de dispositivos únicos, o estágio final é agrupar os dispositivos que pertencem a uma mesma categoria e verificar se o classificador Random Forest mantém bons resultados. Os 27 dispositivos são agrupados em 8 categorias diferentes que podem ser vistas na tabela 6.2. Com as categorias determinadas, 8 novas bases foram montadas, sendo compostas 50% por pacotes de dados de seus respectivos dispositivos e 50% por pacotes da base de dados anômalos.

Os métodos de treinamento e teste e as métricas usadas são os mesmo da classificação por dispositivo único. Os resultados obtidos pelos classificadores nos testes podem ser observados na tabela 6.3 e no gráfico da figura 6.2.

Dispositivo	Acurácia (accuracy)	Precisão (precision)	Revocação (recall)
Aria	0.99948	1.00000	0.99907
D-LinkCam	0.99893	0.99946	0.99840
D-LinkDayCam	1.00000	1.00000	1.00000
D-LinkDoorSensor	0.99671	0.99731	0.99610
D-LinkHomeHub	0.99729	0.99773	0.99685
D-LinkSensor	0.99925	0.99977	0.99875
D-LinkSiren	0.99932	0.99983	0.99881
D-LinkSwitch	0.99938	0.99976	0.99900
D-LinkWaterSensor	0.99955	0.99992	0.99917
EdimaxCam	1.00000	1.00000	1.00000
EdimaxPlug1101W	1.00000	1.00000	1.00000
EdimaxPlug2101W	0.99970	1.00000	0.99935
EdnetCam	1.00000	1.00000	1.00000
EdnetGateway	0.99965	1.00000	0.99931
HomeMaticPlug	1.00000	1.00000	1.00000
HueBridge	0.99859	0.99896	0.99822
HueSwitch	0.99954	0.99967	0.99941
iKettle2	1.00000	1.00000	1.00000
Lightify	1.00000	1.00000	1.00000
MAXGateway	1.00000	1.00000	1.00000
SmarterCoffee	1.00000	1.00000	1.00000
TP-LinkPlugHS100	1.00000	1.00000	1.00000
TP-LinkPlugHS110	1.00000	1.00000	1.00000
WeMoInsightSwitch	0.99959	0.99990	0.99928
WeMoLink	0.99991	1.00000	0.99982
WeMoSwitch	0.99980	0.99987	0.99974
Withings	1.00000	1.00000	1.00000

Tabela 6.1: Resultados dos classificadores com dados anômalos.

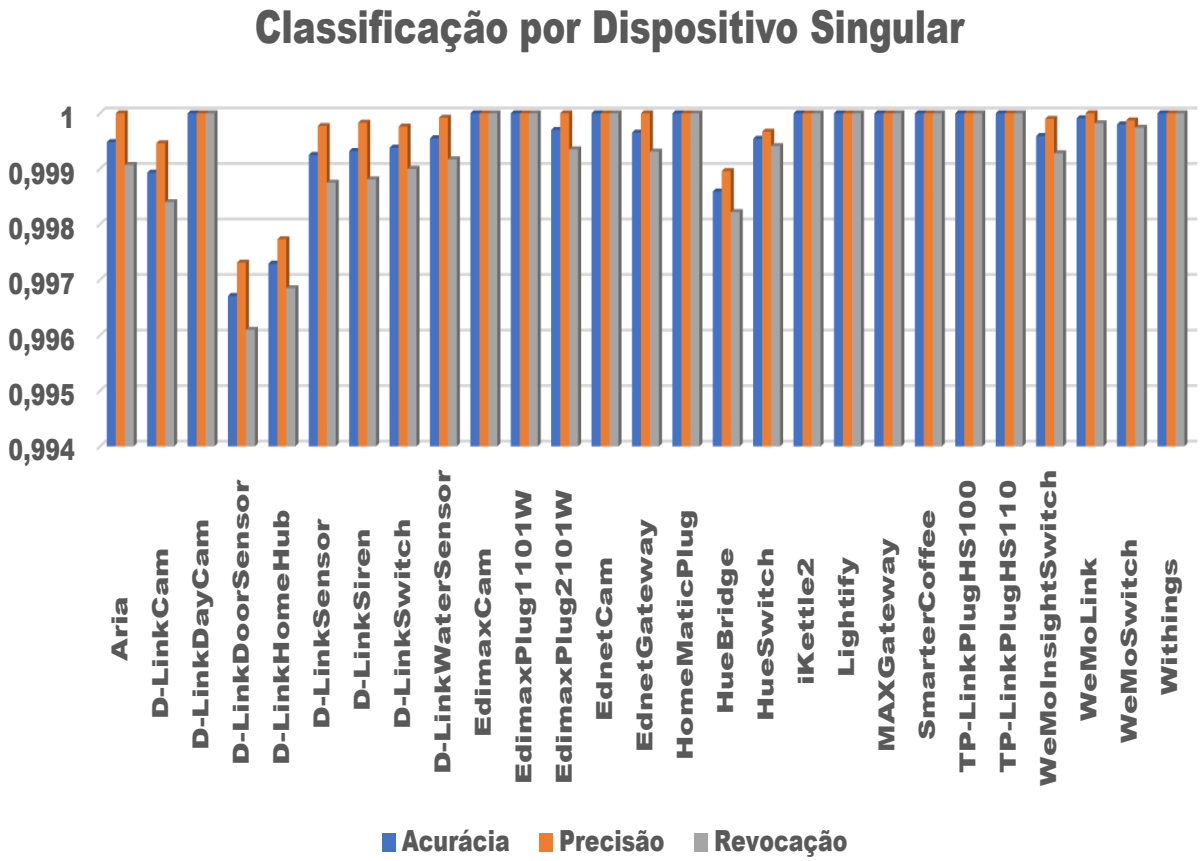


Figura 6.1: Gráfico dos classificadores de dados anômalos por dispositivo. Fonte: O autor.

Categoria	Dispositivos
Câmera smart	D-LinkCam, D-LinkDayCam, EdimaxCam, EdnetCam
Sensor smart	D-LinkDoorSensor, D-LinkSensor, D-LinkWaterSensor
Smart Hub	D-LinkHomeHub, D-LinkSwitch, MAXGateway
Alarme smart	D-LinkSiren
Smart Plug	EdimaxPlug1101W, EdimaxPlug2101W, EdnetGateway, HomeMaticPlug, TP-LinkPlugHS100, TP-LinkPlugHS110, WeMoSwitch
Chaleira smart	iKettle2, SmarterCoffee
Smart light switch	HueBridge, HueSwitch, Lightify, WeMoInsightSwitch, WeMoLink
Balança smart	Aria, Withings

Tabela 6.2: Dispositivos IoT por categoria

Categoria	Acurácia (accuracy)	Precisão (precision)	Revocação (recall)
Câmera smart	0.99920	0.99950	0.99889
Sensor smart	0.99914	0.99937	0.99891
Smart Hub	0.99812	0.99833	0.99790
Alarme smart	0.99945	0.99983	0.99906
Smart Plug	0.99978	1.00000	0.99956
Chaleira smart	1.00000	1.00000	1.00000
Smart light switch	0.99917	0.99930	0.99903
Balança smart	1.00000	1.00000	1.00000

Tabela 6.3: Resultados dos classificadores por categoria de dispositivo com dados anômalos.

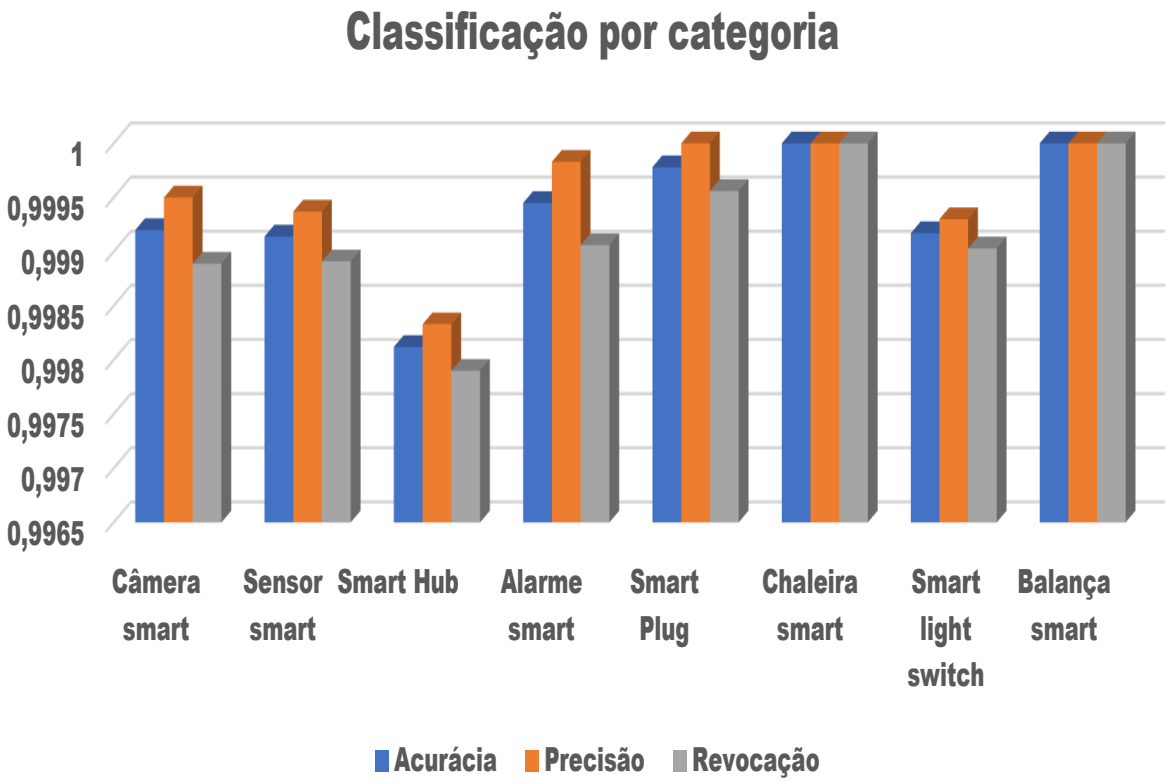


Figura 6.2: Gráfico dos classificadores de dados anômalos por categoria. Fonte: O autor.

As matrizes de confusão geradas por cada classificador categórico com as médias de True Negative, False Negative, True Positive e False Positive, seguindo o padrão apresentado em 4.4 podem ser vistas na tabela 6.4.

Câmera smart	Negative	Positive
Negative	**1005.3**	**0.5**
Positive	**1.1**	**1004.7**

Sensor smart	Negative	Positive
Negative	**2856.7**	**1.8**
Positive	**3.1**	**2855.4**

Smart Hub	Negative	Positive
Negative	**2995.3**	**5.**
Positive	**6.3**	**2994.**

Alarme smart	Negative	Positive
Negative	**1181.1**	**0.2**
Positive	**1.1**	**1180.2**

Smart Plug	Negative	Positive
Negative	**1599.8**	**0.0**
Positive	**0.7**	**1599.1**

Chaleira smart	Negative	Positive
Negative	**47.0**	**0.0**
Positive	**0.0**	**47.0**

Smart light switch	Negative	Positive
Negative	**9408.4**	**6.6**
Positive	**9.1**	**9405.9**

Balança smart	Negative	Positive
Negative	**234.4**	**0.0**
Positive	**0.0**	**234.4**

Tabela 6.4: Matrizes de confusão para cada categoria de dispositivos

6.4 ANÁLISE DOS RESULTADOS

Começando pelos resultados da primeira etapa, na classificação do tipo de dispositivo ao qual o pacote sendo analisado pertence (capítulo 5), os classificadores Random Forest apresentaram as melhores métricas quase que por unanimidade. Ao pegarmos a métrica de acurácia, na tabela 5.4 é possível observar que apenas para 2 dispositivos o Random Forest não ficou em primeiro lugar, ficando em segundo lugar para os dispositivos EdimaxCam e HueBridge, atrás dos classificadores Multilayer Perceptron e SVM para cada dispositivo respectivamente. Para o dispositivo EdnetCam, o Random Forest e o Multilayer Perceptron ficaram empatados, ambos apresentando resultados iguais. Para o restante dos dispositivos o Random Forest ficou em primeiro lugar, com a melhor acurácia dentre todos os classificadores, sendo por isso utilizado na classificação entre dados IoT comuns e anômalos.

Se comparado ao trabalho de (MIETTINEN et al., 2017), que utiliza a mesma base de dados e o classificador Random Forest para identificar dispositivos conectados à rede através de DFP, mas com outro conjunto de atributos (ver tabela 3.3), os meus resultados de acurácia mínimos foram superiores, sendo o classificador para o dispositivo D-LinkSensor que apresentou resultados de 87.99% de acurácia o pior resultado para o RF, enquanto que para (MIETTINEN et al., 2017) os resultados de acurácia para o mesmo dispositivo apresentaram resultados abaixo de 40%. Para alguns dos outros dispositivos (MIETTINEN et al., 2017) atingiu acurácia de 100%, mas para esses mesmos dispositivos, eu consegui resultados entre 89% e 99% de acurácia, sendo os piores resultados atingidos para os dispositivos HueBridge (89.75% de acurácia)

e D-LinkHomeHub (94.4% de acurácia), com o restante dos classificadores atingindo acurácia acima de 95%.

Na segunda parte do trabalho, os resultados das métricas tanto na classificação de dados por dispositivo individual quanto por categoria de dispositivo foram bastante similares, apresentando valores acima de 99% para todas as métricas (acurácia, precisão e revocação). O Random Forest acabou sendo um ótimo classificador para a diferenciação entre dados IoT comuns e anômalos, apresentando resultados bastante superiores em relação à classificação do tipo de dispositivo apresentada capítulo 5. Isso mostra que há uma grande diferença entre pacotes de dados comuns e anômalos, e que é mais fácil para os classificadores Random Forest diferenciar entre estes pacotes do que entre pacotes de dispositivos distintos.

Muitos fatores contribuiram para estes resultados, desde o tamanho da base de dados disponível para cada classificador até o tratamento dos atributos. Talvez o que mais tenha impactado nos resultados tenham sido os atributos base escolhidos para o DFP, apresentados na tabela 4.1. Pois, como visto na tabela 6.4, mesmo para classificadores que apresentavam uma quantidade de dados disponíveis bastante desiguais, como o *Smart light switch* e *Chaleira smart*, a diferença nos resultados não foi tão grande assim, apresentando resultados de acurácia de 99.91% e 100% respectivamente, com o classificador *Chaleira smart* que apresentava uma base menor apresentado resultados levemente superiores. Por isso acredito que as características escolhidas para DFP tenham sido o grande diferencial para os resultados obtidos.

7 CONSIDERAÇÕES FINAIS

O presente trabalho teve por objetivo usar técnicas de Machine Learning para fazer a classificação e identificação correta dos dados de tráfego de redes IoT. Ele explorou tanto a classificação de tráfego entre dispositivos distintos quanto entre pacotes de tráfego comum e anômalo. Para isso, a técnica de Device Fingerprint (DFP) foi utilizada, fazendo a identificação de dispositivos IoT através de amostras de pacotes de rede dos mesmos. Para fazer a classificação de tráfego foram feitas análises de quais características de rede e algoritmos de machine learning são mais comumente utilizados na área de classificação de dispostivos IoT e identificação de tráfego málicioso.

Na parte de classificação de dispostivos, os algoritmos Multilayer Perceptron, K-nearest Neighbors, Support Vector Machine e Random Forest foram testados e ranqueados de acordo com as métricas de Acurácia, Precisão e Revocação, além do teste estatístico de Friedman e Nemenyi. Após os testes, o classificador Random Forest tendo apresentado os melhores resultados foi escolhido para fazer a classificação entre tráfego comum e anômalo.

Para a classificação entre dados de tráfego comum e malicioso foram feitos teste separados com classificadores por dispositivo e por categoria de dispositivo, onde ambos apresentaram uma grande taxa de acerto na diferenciação entre os pacotes de tráfego padrão e anômalos dos dispositivos IoT.

Uma das ideias em relação a trabalhos futuros é fazer uso de atributos derivados, desenvolvendo uma análise do fluxo de pacotes em uma ordem temporal, ao invés de classificar apenas pacotes singulares em ordem aleatória.

Além disso, seria interessante usar alguns algoritmos de Deep Learning, como Convolution Neural Network e Recurrent Neural Network, que apresentaram bons resultados em trabalhos correlacionados, para fazer a classificação dos dados anômalos identificados, fazendo a diferenciação do tipo de ataque detectado.

REFERÊNCIAS

ANEJA, S.; ANEJA, N.; ISLAM, M. S. IoT Device Fingerprint using Deep Learning. , https://arxiv.org/abs/1902.01926, 2018.

BEZAWADA, B. et al. Iotsense: behavioral fingerprinting of iot devices. **arXiv preprint arXiv:1804.03852**, https://arxiv.org/abs/1804.03852, 2018.

GARRETT, T. et al. Traffic Differentiation on Internet of Things. **2018 IEEE Symposium on Service-Oriented System Engineering**, https://ieeexplore.ieee.org/document/8359159, 2018.

GERON, A. **Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems**. https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/: O'Reilly Media, Inc., 2019.

GRANATYR, J. **Modelo afetivo de Reputação utilizando Personalidade e Emoção**: aspectos afetivos aplicados em modelos de confiança e reputação. 2017. Tese (Doutorado em Ciência da Computação) — Pontifícia Universidade Católica do Paraná.

KORONIOTIS, N. et al. Towards the Development of Realistic Botnet Dataset in the Internet of Things for Network Forensic Analytics: bot-iot dataset. , https://arxiv.org/abs/1811.00701, 2018.

KUROSE, J. F.; ROSS, K. W. **Computer Networking**: a top-down approach (6th edition). pearson, 6th edition. https://www.pearson.com/us/higher-education/product/Kurose-Computer-Networking-A-Top-Down-Approach-6th-Edition/9780132856201.html: Pearson, 6th edition., 2012.

MA, Z. et al. Understanding the Mirai Botnet. **26th USENIX Security Symposium**, https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis, 2017.

MIETTINEN, M. et al. IoT Sentinel: automated device-type identification for security enforcement in iot. **2017 IEEE 37th International Conference on Distributed Computing Systems**, https://ieeexplore.ieee.org/document/7980167, 2017.

NGUYEN, T. T.; ARMITAGE, G. A Survey of Techniques for Internet Traffic Classification using Machine Learning. **IEEE COMMUNICATIONS SURVEYS TUTORIALS, VOL. 10, NO. 4, FOURTH QUARTER 2008**, https://ieeexplore.ieee.org/document/4738466, 2008.

NIKAEIN, N. et al. Simple Traffic Modeling Framework for Machine Type Communication. **The Tenth International Symposium on Wireless Communication Systems**, https://ieeexplore.ieee.org/document/6629847, 2013.

PETERSON, L. L.; DAVIE, B. S. **Computer Networks, Fifth Edition**: a systems approach. https://cseweb.ucsd.edu/∼gmporter/classes/wi19/cse124/courseoverview/compnetworks.pdf: Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 5th edition., 2011.

SANTOS, A. C. F. dos. **Uma metodologia para caracterização do tráfego de redes de computadores**. 2011. Tese (Doutorado em Ciência da Computação) — Instituto Nacional de Pesquisas Espaciais - INPE.

SANTOS, B. P. et al. Internet das coisas: da teoria à prática. **Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos**, https://homepages.dcc.ufmg.br/∼mmvieira/cc/papers/internet-das-coisas.pdf, 2016.

SANTOS, M. R. P.; CALLADO, A. d. C. An Architecture Proposal for Network Traffic Monitoring with IoT Traffic Classification Support. **2017 IEEE First Summer School on Smart Cities Natal, Brazil, August 6-11, 2017**, https://ieeexplore.ieee.org/document/8501367, 2017.

SENRIO. **400,000 Publicly Available IoT Devices Vulnerable to Single Flaw.** [Online; accessed 5-August-2019]. https://blog.senr.io/blog/400000-publicly-available-iot-devices-vulnerable-to-single-flaw/.

SENRIO. **Lateral Attacks Between IoT Devices**: the technical details. [Online; accessed 5-August-2019]. https://blog.senr.io/blog/lateral-attacks-between-iot-devices-the-technical-details/.

SENRIO. **Consumer IoT on Enterprise Networks.** [Online; accessed 5-August-2019]. https://blog.senr.io/blog/consumer-iot-on-enterprise-networks/.

SHAHID, M. R. et al. Machine Learning for IoT Network Monitoring. **Institut Mines-Télécom**, https://ressi2019.sciencesconf.org/251004/document, 2019.

SHAIKH, F. et al. A Machine Learning Model for Classifying Unsolicited IoT Devices by Observing Network Telescopes. **IEEE International Wireless Communications and Mobile Computing Conference**, https://ieeexplore.ieee.org/abstract/document/8450404, 2018.

TANENBAUM, A. S.; WETHERALL, D. J. **Computer Networks**. https://www.oreilly.com/library/view/computer-networks-fourth/0130661023/: Prentice Hall Professional Technical Reference, 4th edition., 2002.

WANG, X. et al. Automatic Device Classification from Network Traffic Streams of Internet of Things. **2018 IEEE 43rd Conference on Local Computer Networks**, https://arxiv.org/abs/1812.09882, 2018.

WANG, Y. **Automatic Network Traffic Classification**. 2013. Tese (Doutorado em Ciência da Computação) — Deakin University.