INSTITUTO FEDERAL DE SANTA CATARINA

DANIEL TREVISAN TATSCH

Localização *indoor* utilizando *Bluetooth Low Energy* e aprendizado de máquina

São José - SC

julho/2019

Localização *indoor* utilizando *Bluetooth Low Energy* e aprendizado de máquina

Trabalho de conclusão de curso apresentado à Coordenadoria do Curso de Engenharia de Telecomunicações do campus São José do Instituto Federal de Santa Catarina para a obtenção do diploma de Engenheiro de Telecomunicações.

Orientador: Roberto de Matos

Coorientador: Mario de Noronha Neto

São José - SC

julho/2019

Daniel Trevisan Tatsch
Localização *indoor* utilizando *Bluetooth Low Energy* e aprendizado de máquina/ Daniel Trevisan Tatsch. – São José - SC, julho/2019-
68 p. : il. (algumas color.) ; 30 cm.

Orientador: Roberto de Matos
Monografia (Graduação) – Instituto Federal de Santa Catarina – IFSC
Campus São José
Engenharia de Telecomunicações, julho/2019.

1. Localização *indoor*. 2. *Bluetooth Low Energy*. 2. Aprendizado de máquina. I. Roberto de Matos. II. Instituto Federal de Santa Catarina. III. Campus São José. IV. Localização *indoor* utilizando *Bluetooth Low Energy* e aprendizado de máquina

DANIEL TREVISAN TATSCH

Localização *indoor* utilizando *Bluetooth Low Energy* e aprendizado de máquina

Este trabalho foi julgado adequado para obtenção do título de Engenheiro de Telecomunicações, pelo Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina, e aprovado na sua forma final pela comissão avaliadora abaixo indicada.

São José - SC, 10 de julho de 2019:

Roberto de Matos, Dr.
Orientador
Instituto Federal de Santa Catarina

Mario de Noronha Neto, Dr.
Coorientador
Instituto Federal de Santa Catarina

Eraldo Silveira e Silva, Dr.
Instituto Federal de Santa Catarina

Jorge Henrique B. Casagrande, Dr.
Instituto Federal de Santa Catarina

AGRADECIMENTOS

Agradeço a meus pais, Mari e Altair, que sempre estiveram do meu lado, me motivando e fazendo de tudo para que eu pudesse ter um futuro melhor.

Agradeço também a minha tia de coração, Maria de Lourdes Mello, que me acolheu e me fez companhia em praticamente toda a graduação. Minha eterna gratidão à pessoa mais altruísta que já conheci.

Agradeço a meus orientadores Roberto de Matos e Mario de Noronha Neto por todo suporte dado durante a realização deste trabalho.

Agradeço a minha namorada, Ana Paula Silva Tutui, pelo apoio e compreensão durante a realização deste trabalho. Agradeço também aos novos amigos que fiz do começo ao fim dessa jornada. Sem a companhia diária e a troca de conhecimentos com eles, não seria possível concluir este curso.

Por fim, agradeço a todos os professores do IFSC-SJ por todo o conhecimento transmitido e por disporem de um ensino excepcional.

"A educação é a arma mais poderosa que você pode usar para mudar o mundo."
(Nelson Mandela)

RESUMO

A necessidade de localização de objetos ou pessoas não se limita somente à aplicações em ambientes externos, tendo a localização *indoor* grande importância nos cenários de *Internet of Things* (IoT). Este trabalho apresenta testes e análises de diferentes técnicas de aprendizado de máquina associadas à localização *indoor* utilizando o *Bluetooth Low Energy* (BLE). Os conjuntos de dados usados pelos classificadores foram gerados através da comunicação entre um dispositivo final e três *gateways*. Para desenvolver os dispositivos finais utilizou-se a plataforma Arduino Pro Micro e o módulo BLE AT-09, enquanto o código dos *gateways* foi implementado na plataforma ESP-32. O desempenho dos algoritmos de aprendizado de máquina foi verificado variando da quantidade de setores a serem distinguidos no auditório IFSC-SJ, de acordo com a medida de *Received Signal Strength Indication* (RSSI) obtida por cada *gateway*. Com a segmentação do ambiente em 24 setores, foi possível alcançar uma precisão de 99% com os algoritmos *K-Nearest Neighbor* (K-NN) e *Random Forest*. Para alcançar esses resultados, os valores mais discrepantes do conjunto de medidas foram desconsiderados.

Palavras-chave: Localização *indoor*. Bluetooth Low Energy. Aprendizado de máquina.

ABSTRACT

The need to locate objects or people is not limited only to applications in outdoor environments, but also in indoor, which have great importance in the scenarios of Internet of Things (IoT). This work presents tests and analyzes different machine learning techniques associated with indoor location using Bluetooth Low Energy (BLE). The data sets used by the classifiers were generated through communication between one final device and three gateways. To develop the final devices, the Arduino Pro Micro platform and the BLE AT-09 module were used, while the gateways code was implemented on the ESP-32 platform. The performance of the machine learning algorithms was verified with the variation of the number of sectors to be distinguished in the IFSC-SJ auditorium, according to the measure of Received Signal Strength Indication (RSSI) obtained by each gateway. With the segmentation of the environment in 24 sectors, it was possible to achieve a precision of 99% with the algorithms K-NN and Random Forest. To reach these results, the most discrepant values of the dataset were disregarded.

Keywords: Indoor location. Bluetooth Low Energy. Machine learning.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

IoT *Internet of Things* 9

BLE *Bluetooth Low Energy* 9

M2M *Machine-to-Machine* 25

LPWAN *Low Power Wide Area Network* 25

Mbps Mega bits por segundo 36

NFC *Near Field Communication* 26

RFID *Radio Frequency Identification* 21

GPS *Global Positioning System* 21

RSSI *Received Signal Strength Indication* 9

AoA *Angle of Arrival* 13

ToA *Time of Arrival* 26

ToF *Time of Flight* 29

TDoA *Time Difference of Arrival* 13

ULA *Uniform Linear Array* 13

SUMÁRIO

1 INTRODUÇÃO

Atualmente, a busca por sistemas de controle e monitoramento situados no contexto de IoT tem priorizado equipamentos que possuem baixo custo de implementação e consumo energético ínfimo. Isso permite englobar aplicações altamente escaláveis e com métricas aceitáveis em diferentes contextos, como transporte, saúde, automação industrial, ou até o meio ambiente, onde poderia ser feito por exemplo, o monitoramento de lugares propícios a ocorrência de desastres naturais (AL-FUQAHA et al., 2015). Com o intuito de suprir essas demandas, tecnologias de comunicação como o *Bluetooth Low Energy* (BLE), *Radio Frequency Identification* (RFID) e LoRaWAN possuem como principais características a eficiência energética e custo de implementação reduzido.

À medida que soluções envolvendo IoT se difundem, surge a necessidade de se obter resultados mais aprimorados com tecnologias que antes não eram utilizadas para tais finalidades. No caso do BLE por exemplo, havendo uma rede de dispositivos que se comunicam utilizando essa tecnologia, a mesma pode ter como objetivo determinar a localização desses equipamentos em determinados ambientes.

Estimar o posicionamento de pessoas ou objetos em ambientes abertos, denominados *outdoor*, se tornou algo trivial graças ao *Global Positioning System* (GPS). No entanto, segundo Sadowski e Spachos (2018), sua precisão fica comprometida quando aplicado em ambientes internos, chamados de *indoor*, como prédios ou casas, por conta da dependência de linha de visada dos dispositivos com os satélites de GPS. Alguns fatores que desfavorecem a utilização do GPS para localização *indoor* estão ligados também ao consumo de energia e ao custo de aquisição do sensor.

Para Rezende e Ynoguti (2015), o maior desafio de se implementar a localização *indoor* está nas características do próprio ambiente em si, como a existência de móveis, paredes, estruturas da construção ou equipamentos que utilizam de outras tecnologias de comunicação sem fio. Todos esses fatores podem influenciar na comunicação dos nós que compõem a rede. Para indicar com fidelidade a posição de um usuário ou dispositivo em um ambiente interno, a precisão pode chegar a apenas alguns centímetros, visto que com uma diferença mínima outro ambiente poderia ser registrado.

Dependendo da técnica utilizada, a localização *indoor* pode apresentar resultados na forma simbólica, como setores ou ambientes em que os dispositivos podem estar posicionados. De acordo com Rezende e Ynoguti (2015), a forma em que um sistema de localização *indoor* é utilizado varia de acordo com cada cenário em que é aplicado. Em uma loja por exemplo, pode haver o monitoramento da seção em que cada cliente está, sugerindo ofertas de determinados produtos que estão próximos a sua posição. Em

um hospital, saber a localização de médicos e pacientes pode ser crucial em casos de emergência.

O BLE, bem como outras tecnologias de comunicação sem fio de baixo consumo energético e custo de implementação reduzido, busca suprir essa necessidade de localização em ambientes internos. De acordo com Sadowski e Spachos (2018), essas técnicas podem se basear em diversos parâmetros atrelados à comunicação entre os dispositivos presentes em uma rede de sensores. No entanto, técnicas que utilizam o *Received Signal Strength Indication* (RSSI) oferecem uma demanda maior, visto que não é necessário que se tenha um *hardware* específico e complexo para obtenção desse parâmetro, tornando o sistema mais viável economicamente.

Mesmo que medidas baseadas na potência do sinal recebido sofram grandes variações devido ao ruído presente no ambiente e os efeitos causadas pelo multi percurso, Liu et al. (2007) afirma que modelos que utilizam o RSSI podem obter resultados satisfatórios através da aplicação de técnicas de aprendizado de máquina.

Desenvolveu-se neste trabalho, um sistema de localização *indoor* utilizando o parâmetro RSSI proveniente da comunicação entre um dispositivo final e três *gateways* BLE. Os experimentos realizados variaram o número de setores em que se desejava fazer a distinção dentro do auditório do IFSC-SJ. Para minimizar os problemas relacionados ao uso do RSSI, considerou-se o ambiente sem fluxo de pessoas ou obstruções estruturais.

Os testes consistiram em estimar a posição do dispositivo final, transmitindo constantemente suas informações para os *gateways*. A potência dos sinais recebidos por cada um deles foi enviada para um servidor remoto, gerando diferentes conjuntos de dados que serviram de entrada para os algoritmos de aprendizado de máquina *K-Nearest Neighbor* (K-NN), *Random Forest* e *Support Vector Machine* (SVM). Em todos os experimentos foram consideradas 100 amostras de RSSI de cada *gateway*.

Com a divisão do auditório em 24 setores, sem a exclusão dos valores discrepantes, ou *outliers*, observou-se que o algoritmo *Random Forest* apresentou o melhor desempenho, alcançando em média 96,51% de precisão. Com a exclusão dos *outliers* foi possível obter aproximadamente 99% com todos os algoritmos. No terceiro experimento as medidas anteriores foram agrupadas, formando 4 macro setores. Notou-se que neste caso, houve uma grande variação nas precisões obtidas com o SVM, de acordo com a função de *kernel* utilizada. A classificação utilizando a função gaussiana garantiu uma precisão média de 96,81%, enquanto a função linear (padrão) do SVM obteve apenas 64,97%, considerando os *outliers*.

1.1 Objetivo geral

Analisar diferentes técnicas de aprendizado de máquina associadas à localização *indoor*, utilizando como tecnologia de comunicação o *Bluetooth Low Energy* (BLE).

1.2 Objetivos específicos

- Estudar os parâmetros de comunicação utilizando BLE para uso na localização *indoor*;
- Desenvolver protótipos de *gateways* e dispositivos finais BLE;
- Coletar informações dos dispositivos finais para alimentar uma base de dados de acordo com suas posições.

1.3 Organização do texto

Este trabalho está dividido em 3 partes. Primeiramente, no Capítulo 2 são apresentados os conceitos de *Internet of Things* (IoT), seguido dos parâmetros, técnicas e tecnologias utilizadas na localização *indoor*. Por ser a tecnologia escolhida no desenvolvimento do trabalho, a seção 2.5 aborda as características do *Bluetooth Low Energy* (BLE). Por fim, a seção 2.6 define o tema aprendizado de máquina, destacando os algoritmos utilizados neste projeto.

O Capítulo 3 descreve as atividades realizadas no decorrer deste trabalho. Estão presentes nele, o desenvolvimento do *hardware* (*gateways* e dispositivo final) e do *backend* (servidor e banco de dados). Além disso, também é esclarecida a implementação dos *Supervised Learning Algorithms* (SLAs) utilizados nos testes.

O Capítulo 4 apresenta os resultados obtidos em cada um dos cenários de teste implementados. Todos os códigos fonte, bem como o esquema do banco de dados foram disponibilizados na plataforma GitHub[1].

No Capítulo 5 estão presentes as conclusões sobre este trabalho. Nele, o assunto abordado é brevemente retomado, apontando também os resultados obtidos no capítulo anterior. Por fim, possíveis adaptações e abordagens envolvendo a localização *indoor* e os algoritmos de aprendizado de máquina são indicadas para trabalhos futuros.

[1] https://github.com/danieltatsch/TCC

2 FUNDAMENTAÇÃO TEÓRICA

Os tópicos apresentados neste capítulo abordam assuntos considerados relevantes para o desenvolvimento do trabalho. Na seção 2.1, o tema IoT é definido, havendo também uma breve discussão sobre as tecnologias que compõem os seus sistemas. Nas seções 2.2, 2.3 e 2.4 são definidos respectivamente os parâmetros, técnicas e tecnologias de comunicação utilizados para a realização da localização *indoor*. Na seção 2.5 a tecnologia BLE, por ser o foco deste trabalho, é apresentada mais detalhadamente. Por fim, a seção 2.6 se encarrega de definir o conceito de aprendizado de máquina, apresentando também os *Supervised Learning Algorithms* (SLAs) que serão utilizados no desenvolvimento deste projeto.

2.1 Internet das Coisas

O conceito de IoT está ligado a capacidade de diversos objetos, dispositivos ou equipamentos estabelecerem comunicação entre si. Essa forma de interação constitui um tipo de comunicação denominada *Machine-to-Machine* (M2M), que possibilita que diversos sensores se comuniquem tipicamente pelo meio sem fio e troquem informações de forma autônoma (XIAO et al., 2016). Além disso, essa conexão viabiliza a criação de aplicações voltadas a diferentes contextos, como transporte, saúde, automação industrial ou até o meio ambiente, onde poderia ser feito por exemplo, o monitoramento de lugares propícios a ocorrência de desastres naturais (AL-FUQAHA et al., 2015).

Com a possibilidade do desenvolvimento de uma rede de IoT específica para cada um dos cenários apresentados, surge então, a necessidade de analisar e delimitar os requisitos que serão impostos sobre a rede, bem como suas restrições. De acordo com Loureiro et al. (2003), os dispositivos que compõem uma rede de sensores sem fio possuem limitações quanto ao processamento, memória, taxa de transmissão de dados, alcance e consumo de energia. Sendo assim, é evidente que uma única tecnologia de comunicação não seja capaz de suprir com todas essas necessidades.

Segundo Samie, Bauer e Henkel (2016), para aplicações em que se faz necessário ter uma vasta área de cobertura aliada ao baixíssimo consumo energético, tecnologias pertencentes as redes *Low Power Wide Area Network* (LPWAN), como SigFox e LoRaWAN tendem a ser opções viáveis. Por mais que possuam baixas taxas de transmissão, dependendo da distância entre os componentes da rede, essas tecnologias possuem certas vantagens em relação a rede celular, se comparados os custos de implementação e consumo de energia.

Existem também, tecnologias que se encarregam de cobrir áreas ainda menores, como o RFID e o *Near Field Communication* (NFC), tendo cobertura de apenas alguns centímetros e baixas taxas de transmissão. A vantagem dessas tecnologias é prover de elementos passivos, chamados de *tags*, para comunicação. No RFID por exemplo, existe um elemento ativo que envia constantemente um sinal de controle. Quando uma *tag* é aproximada, esse sinal é identificado e rotulado por ela com o seu registro único, sendo então, enviado de volta para o transmissor (AL-FUQAHA et al., 2015).

Para os cenários de IoT que não têm como requisito o longo alcance, pode ser necessário no entanto, obter taxas de transmissão mais elevadas com um *hardware* de custo acessível. Tecnologias como o Bluetooth e BLE possuem como atrativos essas características, tendo o BLE vantagens no que diz respeito ao consumo energético, tempo de configuração e suporte às topologias de rede em estrela com número ilimitado de nós (SAMIE; BAUER; HENKEL, 2016). Nas próximas seções são explicitados os parâmetros de comunicação, bem como técnicas que utilizam desses parâmetros para realizar a localização *indoor*. São apresentadas também, tecnologias de IoT que podem ser utilizadas para essas aplicações.

2.2 Parâmetros da comunicação

Com o objetivo de realizar a implementação de um sistema de localização *indoor*, antes da escolha da técnica a ser utilizada, deve-se analisar primeiramente quais características da comunicação entre os dispositivos da rede serão consideradas. Essa escolha impacta diretamente na complexidade, no custo e no consumo de energia dos equipamentos do sistema. Segundo Liu et al. (2007), os modelos de localização mais comuns são baseados em parâmetros como a potência do sinal recebido, chamada de RSSI; o ângulo de incidência do sinal na antena do receptor, conhecido como AoA; o tempo de chegada do sinal, *Time of Arrival* (ToA), ou o TDoA, que especifica a diferença do tempo de chegada dos sinais na antena do receptor.

2.2.1 *Received Signal Strength Indication* (RSSI)

De acordo com Sadowski e Spachos (2018), o *Received Signal Strength Indication* (RSSI) é o parâmetro mais simples e popular utilizado em técnicas de localização *indoor*. Isso se da pelo fato de que esse dado pode ser obtido sem nenhum incremento de processamento de *hardware*. No entanto, sua precisão fica comprometida por conta da interferência causada pela presença do ruído no sinal propagado. Além disso, o multi percurso elevado nos ambientes internos causado por pessoas e móveis também ocasiona erros na estimativa do posicionamento.

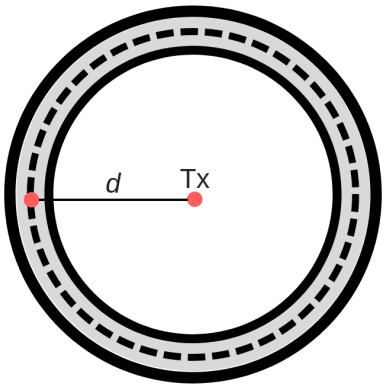
Analisando o fato de que o ruído presente no sinal varia conforme o ambiente em que este sinal é transmitido, sua propagação segue um modelo de perda de caminho conhecido como sombreamento log-normal (RAPPAPORT, 2001). Nele ocorre um decaimento logaritmo do sinal em relação a distância, variando de forma aleatória segundo uma distribuição normal. Sendo assim, pode-se representar a perda de caminho $P(d)$ em função da distância d e da variável aleatória X de média nula e desvio padrão σ através da Equação 2.1.

$$P(d) = P(d_0) - 10n \cdot log_{10}(\frac{d}{d_0}) + X_\sigma \tag{2.1}$$

Em que n é o expoente de perda de caminho que indica a velocidade com a qual o sinal é atenuado, d_0 corresponde a uma distância de referência e $P(d_0)$ indica a potência do sinal em relação a essa distância.

Sendo o RSSI conhecido pelo receptor, a distância d entre ele e o dispositivo transmissor (Tx) pode ser estimada através de um círculo de incerteza, conforme mostra a Figura 1. A área em volta do tracejado explicita as imprecisões nas medições do RSSI, que podem causar um deslocamento do valor ideal da distância encontrada.

Figura 1 – Região de incerteza utilizando o RSSI



Fonte: Baseado em Sahinoglu, Gezici e Güvenc (2008).

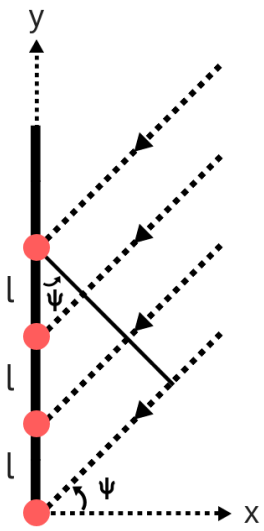
Utilizar o RSSI como parâmetro para determinar a localização *indoor* se torna viável graças a facilidade na obtenção de resultados e o baixo custo do *hardware* envolvido. Mesmo que sofra com o ruído e com a atenuação devido ao multi percurso, modelos que utilizam esse parâmetro podem obter melhores resultados realizando múltiplas medições com vários pontos conhecidos e aplicando de técnicas de aprendizado de máquina (LIU et al., 2007).

2.2.2 Angle of Arrival (AoA)

Para os modelos baseadas no *Angle of Arrival* (AoA), é necessário que se tenha um arranjo de antenas para determinar o ângulo no qual o sinal é propagado. Consequentemente, esses modelos possuem um custo maior, pois requerem um *hardware* complexo e calibragens para que se obtenha uma precisão satisfatória (SADOWSKI; SPACHOS, 2018).

Segundo Sahinoglu, Gezici e Güvenc (2008), um dos arranjos de antenas mais comuns é chamado de *Uniform Linear Array* (ULA) e consiste no alinhamento e espaçamento uniforme entre todas as antenas, conforme mostra a Figura 2, onde os pontos no eixo y representam as antenas receptoras, espaçadas por uma distância l. Com isso, o ângulo de chegada ψ de um sinal pode ser obtido de acordo com os atrasos subsequentes em cada antena. Por outro lado, para sinais de banda estreita, podem ser utilizadas as diferenças de fase entre os sinais que chegam nas antenas do arranjo, podendo ser combinadas a fim de estimar a direção de origem do sinal recebido.

Figura 2 – *Uniform Linear Array* (ULA)



Fonte: Baseado em Sahinoglu, Gezici e Güvenc (2008).

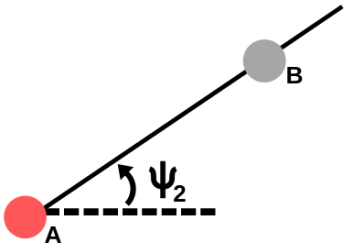
O cálculo do AoA pode ser realizado conhecendo o atraso de propagação τ na antena subsequente, como mostra a Equação 2.2

$$\psi = \arcsin\left(\tau \cdot l \cdot c\right) \tag{2.2}$$

Sendo ψ o valor do AoA, l o espaçamento entre as antenas e c a velocidade da luz.

Diferente do que ocorre com o RSSI, em que a distância entre dois pontos pode ser obtida, a Figura 3 mostra que a medida do AoA determina uma direção da origem do sinal recebido de acordo com o ângulo ψ formado entre o ponto de referência A e o transmissor B. (SAHINOGLU; GEZICI; GÜVENC, 2008).

Figura 3 – Direção da origem do sinal recebido com o AoA



Fonte: Baseado em Sahinoglu, Gezici e Güvenc (2008).

2.2.3 Time of Arrival (ToA)

O parâmetro *Time of Arrival* (ToA), também conhecido como *Time of Flight* (ToF), é utilizado em sistemas que baseiam-se na sincronização dos tempos de envio e chegada de um sinal entre transmissor e receptor para estipular a distância entre os dois pontos (FARID; NORDIN; ISMAIL, 2013). O resultado encontrado é análogo ao apresentado na Figura 1, e pode ser estimado através do produto entre tempo de propagação do sinal até o receptor e a velocidade da luz.

Para Gu, Lo e Niemegeers (2009), técnicas que utilizam desse parâmetro para realizar a localização *indoor* de dispositivos são consideradas as mais precisas, podendo filtrar os efeitos de multi percurso presentes nos ambientes internos. Porém, segundo Farid, Nordin e Ismail (2013), apesar da precisão, os métodos que utilizam ToA necessitam de equipamentos específicos para estabelecer com fidelidade o sincronismo entre todos os dispositivos da rede, aumentando assim o custo do sistema.

2.2.4 Time Difference of Arrival (TDoA)

Diferente do ToA que requer sincronização entre os dispositivos de referência e os que se deseja estimar a localização, as técnicas que utilizam o *Time Difference of Arrival* (TDoA) necessitam apenas da sincronização entre os pontos de referência (FARID; NORDIN; ISMAIL, 2013).

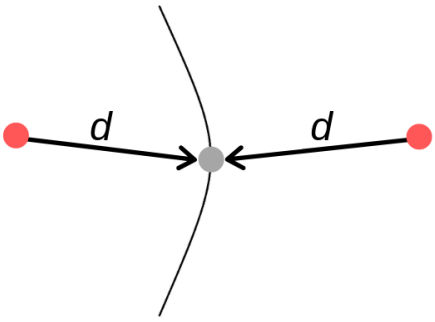
De acordo com Sahinoglu, Gezici e Güvenc (2008), uma forma de obter o TDoA é estimar o ToA em um receptor para dois dos pontos de referência, realizando então a diferença entre esses valores, conforme apresentado na Equação 2.3

$$\tau = \tau_1 - \tau_2 \tag{2.3}$$

Sendo τ_1 e τ_2 as representações da estimativa do ToA do primeiro e segundo pontos de referência, respectivamente.

Com essa diferença de tempo calculada, o dispositivo alvo fica posicionado em uma região de incerteza formada por uma hipérbole, conforme mostra a Figura 4.

Figura 4 – Região de incerteza utilizando o TDoA



Fonte: Baseado em Sahinoglu, Gezici e Güvenc (2008).

2.3 Técnicas de localização

Nesta subseção são apresentadas técnicas de localização *indoor* que se baseiam em informações do dispositivo a ser localizado, relacionadas aos demais componentes da rede. Para tal, são utilizados os parâmetros apresentados na subseção 2.2.

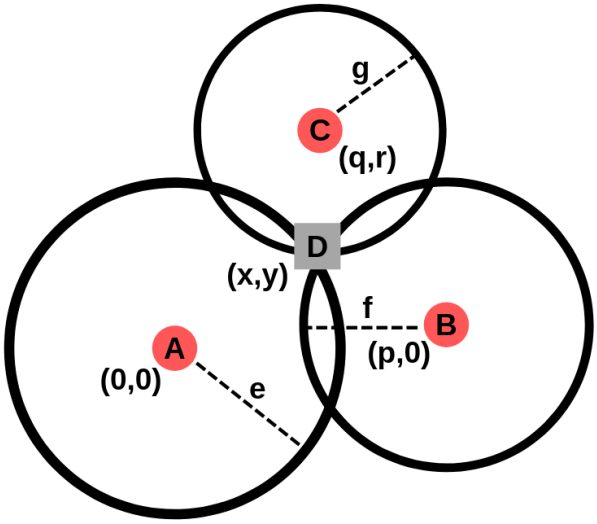
Primeiramente são apresentadas técnicas que utilizam abordagens geométricas para determinar a localização de um dispositivo, através da interseção das posições de um conjunto de pontos de referência. Por fim, é apresentada a técnica utilizada neste trabalho, chamada de *Fingerprinting*, que parte de modelos estatísticos para determinar a localização *indoor*.

2.3.1 Trilateração

O método da trilateração consiste na obtenção da localização em duas dimensões de um dispositivo através da sua distância em relação a três pontos de referência, onde suas posições são conhecidas (SADOWSKI; SPACHOS, 2018).

Parâmetros como o RSSI ou o ToA podem ser utilizados para estabelecer a distância entre cada um dos pontos e o dispositivo alvo. Com isso, uma região de incerteza é formada de acordo com cada uma das distâncias estimadas. A Figura 5 apresenta a região de incerteza D com coordenadas $(x; y)$ formadas pela interseção dos pontos A, B e C, que possuem coordenadas $(0; 0)$, $(p; 0)$ e $(q; r)$ e distâncias e, f e g, respectivamente.

Figura 5 – Região de incerteza utilizando a trilateração



Fonte: Baseado em Sadowski e Spachos (2018).

Sendo conhecidas as distâncias que correspondem aos raios das circunferências na Figura 5, as coordenadas do dispositivo alvo podem ser determinadas através das equações 2.4 e 2.5 apontadas por Sadowski e Spachos (2018).

$$x = \frac{e^2 - f^2 + p^2}{2 \cdot p} \tag{2.4}$$

$$y = \frac{e^2 - g^2 + q^2 + r^2}{2 \cdot r} - x \cdot \frac{q}{r} \tag{2.5}$$

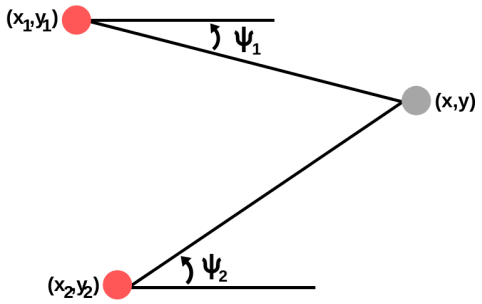
2.3.2 Triangulação

Segundo Sahinoglu, Gezici e Güvenc (2008), a triangulação tem como objetivo estimar a localização *indoor* através do AoA entre apenas dois dispositivos de referência e o que se deseja obter a posição, desde que este não esteja alinhado com os pontos conhecidos. A Figura 6 ilustra os dois pontos de referência com as coordenadas $(x1; y1)$ e $(x2; y2)$, respectivamente, em que $\psi 1$ e $\psi 2$ correspondem aos ângulos de incidência do sinal transmitido pelo dispositivo de interesse $(x; y)$. Suas coordenadas podem ser calculadas através das equações 2.6 e 2.7.

$$x = \frac{x_2 \cdot \tan\psi_2 - x_1 \cdot \tan\psi_1 + y_1 - y_2}{\tan\psi_2 - \tan\psi_1} \tag{2.6}$$

$$y = \frac{(x_2 - x_1) \cdot \tan\psi_1 \cdot \tan\psi_2 + y_1 \cdot \tan\psi_2 - y_2 \cdot \tan\psi_1}{\tan\psi_2 - \tan\psi_1} \tag{2.7}$$

Figura 6 – Estimativa da localização utilizando a triangulação

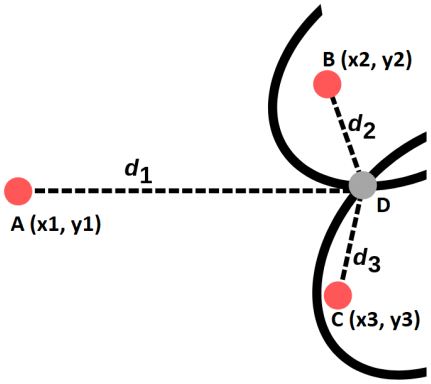


Fonte: Baseado em Sahinoglu, Gezici e Güvenc (2008).

2.3.3 Multilateração

A técnica da multilateração utiliza como parâmetro o TDoA e consiste em agregar no mínimo três dispositivos cuja suas posições são conhecidas (SAHINOGLU; GEZICI; GÜVENC, 2008). A Figura 7 explicita a região de incerteza gerada através dos pontos A $(x1; y1)$, B $(x2; y2)$ e C $(x3; y3)$ e o ponto de interseção D.

Figura 7 – Região de incerteza utilizando a multilateração



Fonte: Baseado em Sahinoglu, Gezici e Güvenc (2008).

A estimativa da localização pode ser realizada então, calculando as regiões das duas hipérboles em relação à distância d_1 através da Equação 2.8 (SAHINOGLU; GEZICI; GÜVENC, 2008).

$$d_{i1} = d_i - d_1 = \sqrt{(x - x_i)^2 + (y - y_i)^2} - \sqrt{(x - x_1)^2 + (y - y_1)^2} \tag{2.8}$$

Para $i = 2, 3$.

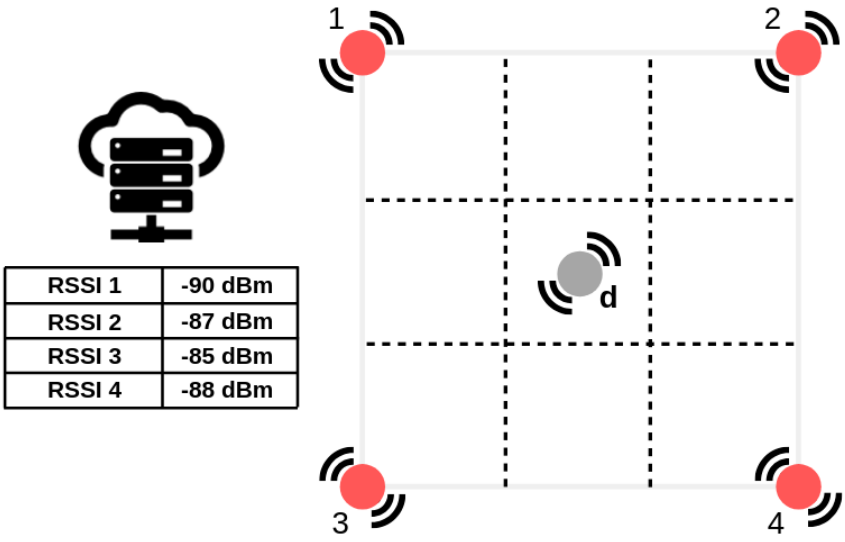
2.3.4 *Fingerprinting*

Apesar das imprecisões associadas à medição do RSSI destacadas na subseção 2.2.1, a técnica *fingerprinting* é uma solução viável para se modelar ambientes que utilizam desse parâmetro para realizar a localização *indoor* (PU; YOU, 2018). Além disso, por estimar a posição dos dispositivos utilizando apenas RSSI, não é necessário utilizar *hardwares* com sincronização de tempo ou arranjos de antenas, diminuindo significativamente o custo e complexidade do sistema.

Segundo Farid, Nordin e Ismail (2013), a localização utilizando *fingerprinting* consiste em duas etapas: calibração ou treino e fase de localização ou testes. A ideia é separar um ambiente em diversos setores, realizando na primeira fase, um mapeamento dessas regiões através da medida do RSSI de diferentes dispositivos conhecidos em relação ao que se deseja analisar. Após a obtenção desses valores em todos os setores, os resultados são armazenados em uma base de dados. Com isso, a segunda etapa consiste em determinar a localização de um dispositivo através da comparação entre os dados atuais e os que foram previamente obtidos na primeira fase. Para manter a precisão desse método, a base de dados gerada na fase de treino deve ser periodicamente atualizada.

A Figura 8 apresenta o processo de localização *indoor* utilizando a técnica *fingerprinting*. Os pontos 1, 2, 3 e 4 correspondem aos dispositivos conhecidos que esperam os dados enviados constantemente em *broadcast* pelo ponto desconhecido *d*. Eles então, alimentam a base de dados na primeira etapa (salvando os valores de RSSI de cada ponto conhecido, por exemplo), ou à utilizam juntamente com as informações capturadas para estimar o setor em que o transmissor se encontra na segunda etapa.

Figura 8 – Cenário de localização utilizando o *fingerprint*



Fonte: Desenvolvida pelo autor.

É importante salientar que não há a necessidade de especificar os pontos de referência através de coordenadas $(x; y)$ como na abordagem geométrica, pois essa técnica retorna como resultado o setor mais provável em que o dispositivo alvo se encontra.

Pelo fato do *fingerprinting* realizar análises estatísticas para determinar a localização *indoor*, na seção seção 2.6 são apresentados diferentes algoritmos de aprendizado de máquina que podem ser combinados com essa técnica para realizar a localização *indoor*.

2.4 Tecnologias utilizadas em sistemas de localização *indoor*

Sistemas de localização *indoor* baseados na tecnologia Wi-Fi (padrões definidos pelo *Institute of Electrical and Electronic Engineers* (IEEE) 802.11) possuem vantagens se comparados a outros modelos, por utilizarem uma tecnologia muito difundida e compatível com praticamente todos os dispositivos que a possuem. Com isso e por utilizar o RSSI, os custos com o *hardware* e a manipulação dos dados são inferiores se comparados a outras tecnologias (FARID; NORDIN; ISMAIL, 2013). Porém, os dispositivos que utilizam o Wi-Fi possuem alto consumo energético, além de sofrerem possíveis interferências por conta de outros equipamentos que operam na mesma faixa de frequência dos 2.4GHz.

A tecnologia *Radio Frequency Identification* (RFID) é usada na localização *indoor* de ambientes complexos e de difícil modelagem, como escritórios e hospitais (GU; LO; NIEMEGEERS, 2009). Utilizando nos receptores (dispositivos que serão localizados) *tags* passivas, ou seja, que não necessitam de uma fonte de alimentação própria, o sistema pode se tornar mais barato. No entanto, o alcance delas é muito pequeno se comparadas às *tags* ativas, que se encarregam de transmitir suas informações e cobrir uma região maior.

Apesar de possuir dispositivos muito pequenos utilizados na localização de pessoas, Gu, Lo e Niemegeers (2009) afirma que para se utilizar técnicas de localização precisas são necessários muitos dispositivos RFID, fazendo com que o custo de um projeto utilizando essa tecnologia seja elevado.

Desenvolvida pela LoRa Alliance, a tecnologia LoRaWAN possui como principal característica o longo alcance aliado a um consumo energético ínfimo. Segundo Sadowski e Spachos (2018), a vantagem em utilizá-la está no fato dela operar na frequência de 915MHz, não sofrendo tanto com interferências causadas por outras tecnologias de outros dispositivos como no caso do Wi-Fi. Porém, um ponto negativo no uso da LoRaWAN está na necessidade de um *hardware* específico, patenteado pela empresa Semtech Corporation. Com isso, o custo de implantação de um sistema de localização *indoor* com LoraWAN se torna elevado.

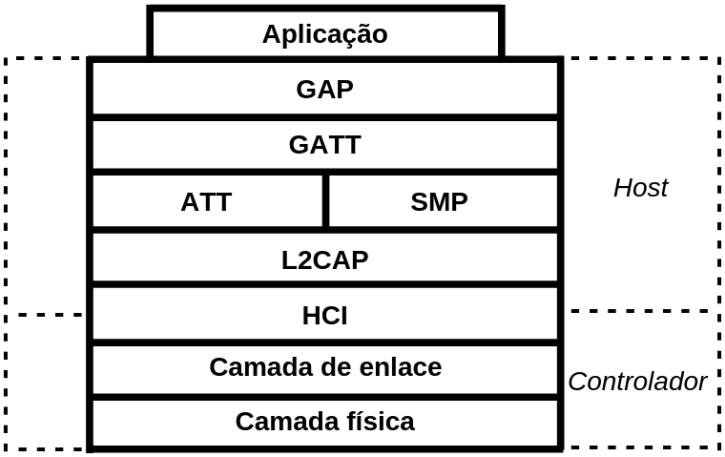
O *Bluetooth Low Energy* (BLE) tem sido uma opção viável na localização *indoor*, pois apesar de não possuir grande alcance e sofrer com as atenuações recorrentes dos ambientes internos, possui um custo menor se comparado às tecnologias Wi-Fi e LoRaWAN, além de um consumo energético ínfimo: em média 1 μA e 15mA de pico em momentos de conexão ou envio de dados (GOMEZ; OLLER; PARADELLS, 2012). Por ser o foco deste trabalho, a seção 2.5 destina-se a apresentar em maiores detalhes essa tecnologia.

2.5 Bluetooth Low Energy

O *Bluetooth Low Energy* (BLE) é uma tecnologia pertencente ao padrão Bluetooth 4.0, tendo sua origem no ano de 2010 através do *Bluetooth Special Interest Group* (SIG). O BLE tem como principal característica o baixo consumo energético associado à aplicações que necessitam apenas de uma baixa taxa de transmissão de dados, contrapondo os cenários em que a sua versão clássica atua. Além disso, o BLE opera sobre um pequeno número de interações entre os componentes da rede, como o envio de valores de sensores ou pequenas mensagens de controle com uma quantidade mínima de cabeçalho (ČABARKAPA; GRUJIć; PAVLOVIć, 2015).

Segundo Darroudi e Gomez (2017), o BLE atua sob uma pilha de protocolos que operam em duas partes: controlador e *host*. O controlador é responsável pela camada física e de enlace. O *host* oferece suporte às camadas superiores para interações entre aplicações, tendo seus protocolos definidos nas subseções 2.5.3 à 2.5.6. A Figura 9 apresenta a estrutura da pilha protocolar do BLE, em que a comunicação entre essas duas camadas é padronizada pelo *Host Controller Interface* (HCI).

Figura 9 – Pilha protocolar do BLE



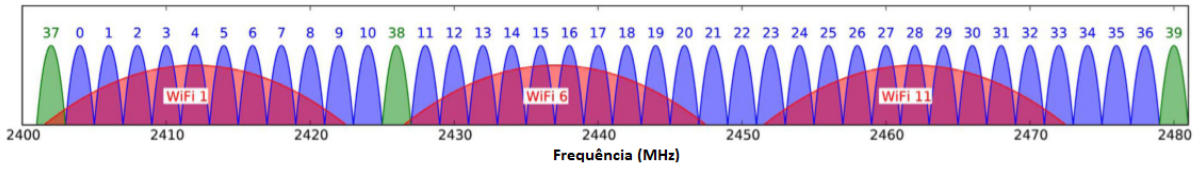
Fonte: Baseado em Chang (2014).

2.5.1 Camada Física

O BLE trabalha na frequência de 2.4GHz, definindo 40 canais de 2MHz cada ao longo do espectro, onde 3 destes são chamados de canais de anúncio e o restante de dados. Os padrões anteriores do *Bluetooth* operam com 79 canais, sendo 32 de anúncio e o restante de dados. Os canais de anúncio têm como finalidade descobrir dispositivos, estabelecer conexões e realizar transmissões em *broadcast*. Os canais de dados ficam responsáveis por realizar a comunicação bidirecional entre os dispositivos conectados (ČABARKAPA; GRUJIć; PAVLOVIć, 2015).

Por operar na mesma faixa de frequência que outras tecnologias, como o Wi-Fi, o BLE tende a sofrer grandes interferências. Para minimizar esse problema, de acordo com Texas Instruments (2016), os canais 37, 38 e 39 foram escolhidos para serem os de anúncio por não sobreporem os canais mais comuns do Wi-Fi: 1, 6 e 11. A Figura 10 ilustra os respectivos canais do BLE, bem como a faixa de frequência em cada um está localizado. Além disso, ela destaca o posicionamento dos canais de anúncio (em verde) em relação aos canais do Wi-Fi (vermelho).

Figura 10 – Canais BLE e sobreposição dos canais do Wi-Fi



Fonte: Baseado em Pu e You (2018).

Possuindo uma taxa de transmissão de dados de até 1Mega bits por segundo (Mbps) e utilizando o esquema de modulação *Gaussian Frequnecy Shift Keying* (GFSK), Čabarkapa, Grujić e Pavlović (2015) afirma que o BLE consegue consumir menos energia por conta do índice de modulação utilizado, alcançando distâncias de até 100 metros com potências de transmissão de -20 a 10dBm e sensibilidade dos dispositivos receptores em torno de -93dBm (GOMEZ; OLLER; PARADELLS, 2012). Além disso, é utilizada a técnica *Adaptative Frequency Hopping* (AFH) nos canais de dados, que determina qual dos 37 canais será utilizado para comunicação durante um tempo pré estabelecido, alternado entre os canais de forma pseudo-aleatória. Essa técnica ajuda a diminuir a possibilidade de interferências e desvanecimento causados pela existência de outras tecnologias que operam na mesma faixa de frequência que o BLE.

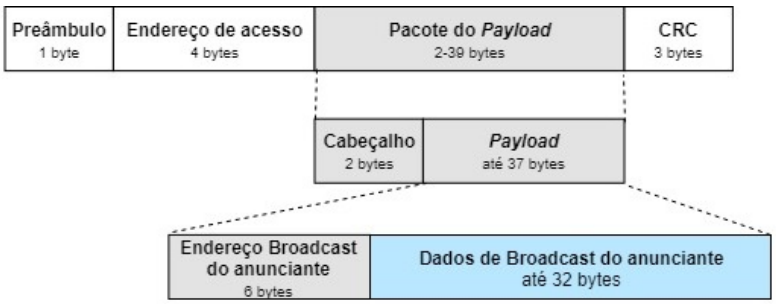
2.5.2 Camada de Enlace

De acordo com Čabarkapa, Grujić e Pavlović (2015), a comunicação entre dois ou mais dispositivos BLE pode ser feita seguindo dois tipos de padrões estabelecidos pela camada de enlace. No primeiro, os dispositivos se comportam como anunciante e *scanner*. O anunciante, como o nome sugere, se aproveita dos canais de anúncio para transmitir seus dados em *broadcast*, enquanto o *scanner* fica basicamente esperando alguma informação.

O segundo padrão a ser utilizado especifica que haja uma conexão entre o anunciante e o *scanner*, estabelecendo características de mestre e escravo, respectivamente. Através da técnica de acesso ao meio *Time Division Multiple Access* (TDMA), o mestre pode estabelecer conexão com diversos escravos, sendo que o escravo pode se conectar com apenas um mestre por vez. Essa arquitetura de comunicação constitui a chamada *piconet*, que segue uma topologia de rede em estrela (GOMEZ; OLLER; PARADELLS, 2012).

A Figura 11 mostra o formato de um quadro de anúncio do BLE, sendo o byte de preâmbulo utilizado para sincronização, seguido por 4 bytes de endereço de acesso, mais 2 à 39 bytes de informação (sendo 2 o tamanho do cabeçalho fixo) e por fim o *Cyclic Redundancy Check* (CRC), responsável por detectar possíveis erros na recepção do pacote.

Figura 11 – Formato do quadro de anúncio do BLE



Fonte: Baseado em Texas Instruments (2016).

Ao se utilizar o primeiro padrão definido anteriormente, os dados que o anunciante envia não podem ser maiores que 32 dos 37 bytes de *payload* real do BLE, sendo que o seu endereço é enviado nos 6 bytes que antecedem os dados de *broadcast*.

Segundo Chang (2014), o fato do quadro BLE possuir poucos bytes, atribui à tecnologia a possibilidade de estabelecer conexões de forma rápida, transmitindo o pacote com a carga máxima de informações no pacote do *payload* em 300 μs.

2.5.3 Logical Link Control and Adaption Protocol (L2CAP)

Sendo uma versão otimizada e simplificada do mesmo protocolo utilizado no Bluetooth clássico, o *Logical Link Control and Adaption Protocol* (L2CAP) tem como objetivo multiplexar os dados dos protocolos superiores e inferiores, permitindo uma comunicação lógica entre os dispositivos (GOMEZ; OLLER; PARADELLS, 2012). Esse processo não faz uso de mecanismos de retransmissão ou controle de fluxo, presentes nas outras versões do Bluetooth.

2.5.4 Attribute Protocol (ATT) e Generic Attribute Profile (GATT)

O *Attribute Protocol* (ATT) é responsável por desempenhar funções de cliente e servidor em uma comunicação entre dois dispositivos. O servidor mantém uma estrutura de dados chamada de atributo, que é encapsulada e gerenciada pelo protocolo superior da pilha, o *Generic Attribute Profile* (GATT). O cliente pode acessar os atributos do servidor (modificando ou obtendo valores) através de solicitações e respostas ou indicações e confirmações (GOMEZ; OLLER; PARADELLS, 2012).

O *Generic Attribute Profile* (GATT) é responsável por definir uma estrutura que, a partir do ATT, realiza a descoberta de serviços e o envio ou requisição de características entre os dispositivos. Uma característica corresponde a um conjunto de dados, incluindo valores e propriedades. Gomez, Oller e Paradells (2012) especifica que tanto as características quanto os serviços são especificados através dos atributos do ATT, como por exemplo: um servidor que executa um serviço “sensor de temperatura” pode possuir uma característica “temperatura”, que por sua vez utiliza de atributos para descrever o sensor, armazenar os valores adquiridos e especificar as unidades de medida.

2.5.5 Generic Access Profile (GAP)

O *Generic Access Profile* (GAP) está presente no topo da pilha protocolar do BLE, sendo assim, seu papel é especificar os padrões e procedimentos que as aplicações necessitam para realizar a descoberta de dispositivos, o gerenciamento da conexão e a segurança.

São definidos nesse protocolo quatro padrões com requisitos específicos: *broadcaster*, *observer*, periférico e central. Os dois primeiros padrões correspondem ao envio e recepção de dados sem que aconteça o estabelecimento de conexão entre os dispositivos. No padrão estabelecido na central, o dispositivo tem como característica iniciar e gerenciar diversas conexões, enquanto o periférico é projetado para dispositivos que apenas necessitam se conectar a uma central. Esse comportamento entre a central e o periférico explicita a relação mestre e escravo definida na subseção 2.5.2.

2.5.6 Security Manager Protocol (SMP)

Segundo Chang (2014), o *Security Manager Protocol* (SMP) é responsável pelo pareamento dos dispositivos e distribuição de chaves de segurança para garantir a identidade e criptografia de dados. Além disso, ele se encarrega de diminuir a carga de trabalho dos dispositivos escravos, por estes não possuírem um poder de processamento equivalente ao do mestre.

Para cumprir com os objetivos desse trabalho, são necessários apenas os parâmetros da comunicação entre os dispositivos, sem precisar adquirir dados de sensores específicos. Sendo assim, optou-se por realizar o padrão anunciante e *scanner* especificado na subseção 2.5.2. Com isso, os anunciantes apenas enviam suas identificações, que por sua vez são capturadas pelos *scanners* juntamente com o RSSI, sem a necessidade de se estabelecer conexão entre cada um dos dispositivos.

2.6 Aprendizado de Máquina

Aprendizado de máquina é um segmento da Inteligência Artificial (IA) que tem como objetivo construir algoritmos capazes de construir fluxos lógicos de decisão de forma automatizada. Para tal, geram modelos de previsão e análise de dados que se baseiam em resultados conclusivos obtidos através de experiências anteriores (MONARD; BARANAUSKAS, 2003).

Segundo Qiu et al. (2016), diversas áreas, como medicina, astronomia e biologia, que geram uma quantidade complexa e massiva de dados, têm utilizado com frequência técnicas de aprendizado de máquina. Isto se da pelo fato de que estas técnicas podem fornecer possíveis soluções para a obtenção de informações escondidas entre uma enorme quantidade de dados.

Além disso, o aprendizado de máquina tem sido considerado a principal tecnologia para gerenciamento autônomo de redes inteligentes. Cui et al. (2018) afirma que em sistemas de IoT por exemplo, que têm se tornado cada vez mais dinâmicos, heterogêneos e complexos, o uso de aprendizado de máquina permite o desenvolvimento de aplicações mais eficientes. Com isso, é possível prover serviços melhores a fim de se obter mais usuários.

Apesar de ser uma ferramenta poderosa, não existe um método de aprendizado de máquina capaz de representar todo e qualquer sistema. Logo, de acordo com Monard e Baranauskas (2003), é necessário analisar as características de cada uma das técnicas disponíveis, verificando qual algoritmo melhor se encaixa na solução de um determinado problema.

O aprendizado de máquina pode ser dividido em três categorias: supervisionado, não supervisionado e aprendizado de reforço, sendo as duas primeiras mais comumente utilizadas. De acordo com Lovon-Melgarejo et al. (2018), os algoritmos pertencentes ao aprendizado de máquina supervisionado, ou *Supervised Learning Algorithms* (SLAs), consistem basicamente em duas etapas: treino e classificação. Na primeira fase é gerado um modelo de categorização a partir de amostras das características coletadas. Esses dados são utilizados posteriormente para inferir a classificação das novas informações coletadas no sistema durante a segunda fase. Assim, como o resultado é conhecido, é possível estimar o desempenho do classificador.

O aprendizado de máquina não supervisionado é aplicado em modelos que não possuem valores de saída, chamados de descritivos. Os padrões ou comportamentos do sistema são estimados através de um conjunto de dados previamente disponibilizado (ATHMAJA; HANUMANTHAPPA; KAVITHA, 2018). No caso do aprendizado por reforço, os algoritmos são realimentados de acordo com os resultados previamente obtidos. Com o aprendizado constante, esses algoritmos tendem a oferecer comportamentos muito específicos de acordo com novos dados de entrada.

Como neste trabalho a localização *indoor* é baseada no método *fingerprinting*, os modelos de predição utilizados correspondem ao aprendizado de máquina supervisionado. Sendo assim, as subseções a seguir apresentam alguns dos SLAs comumente utilizados, segundo Lantz (2015).

2.6.1 *K-Nearest Neighbor* (K-NN)

De forma geral, os classificadores do tipo *nearest neighbors*, ou vizinhos próximos, têm como objetivo classificar dados desconhecidos (não rotulados) de acordo com suas similaridades com classes previamente definidas. O K-NN é um dos algoritmos que mais utilizam essa abordagem, além de ser um dos mais simples.

O K-NN recebe este nome pelo fato de utilizar informações de K vizinhos próximos para classificar os dados não rotulados. Após a fase de treino, onde vários itens já foram classificados em diversas categorias, o algoritmo identifica K registros nos dados de treinamento que são mais próximos através da similaridade. Com isso, na fase de testes, o dado desconhecido é atribuído à classe da maioria dos K vizinhos próximos.

Lantz (2015) afirma que a escolha de K impacta diretamente no resultado da classificação. Escolhendo um número muito grande é possível reduzir o impacto ou variância causada por dados ruidosos, no entanto, padrões pequenos, mas importantes podem ser desconsiderados. Se um único vizinho for considerado ($K = 1$), pode ocorrer a classificação errônea por conta de dados discrepantes registrados na fase de treino. A escolha do número de vizinhos próximos é realizada então, testando diversos valores para K em vários

conjuntos de dados de treinamento e verificando qual oferece melhor desempenho na classificação.

Para calcular a proximidade entre um dado não rotulado e seus vizinhos próximos, o K-NN utiliza a distância Euclidiana, ou seja, o menor caminho entre os dois pontos. Seu cálculo pode ser realizado através do somatório apresentado na Equação 2.9, onde D é a distância entre p e q, que são os respectivos pontos, com n características. (LANTZ, 2015).

$$D = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{N}(p_i - q_i)^2} \tag{2.9}$$

2.6.2 Árvore de decisão

Classificadores que realizam o algoritmo da árvore de decisão utilizam uma estrutura de ramificação, que começa em um único ponto denominado raiz. Quando um novo dado começa a ser analisado, este é passado pelos pontos de decisão, exigindo a escolha de um determinado caminho, ou ramo, de acordo com os seus atributos. Quando uma decisão final é realizada, a árvore encontra seus nós de folha, também chamados de nós terminais, que fornecem o resultado esperado dado a série de eventos ocorrida.

O benefício de se utilizar a árvore de decisão está no fato de que a sua estrutura não se limita a apenas o uso do algoritmo de aprendizado de máquina. Após a criação do modelo é possível gerar uma estrutura em formato legível por humanos, fornecendo uma visão sobre como e por qual motivo este modelo funciona bem ou mal para uma determinada tarefa.

Através dessas características, Lantz (2015) apresenta alguns exemplos de sistemas que utilizam o algoritmo da árvore de decisão:

- Análise de crédito e pontuações bancárias, nos quais os critérios fazem com que o cliente tenha um empréstimo aprovado ou não;
- Estudo de *marketing* de comportamento do cliente através de pesquisas de satisfação, sendo compartilhadas com agências de publicidades;
- Diagnóstico de condições médicas com base em medições laboratoriais, sintomas ou taxa de progressão da doença.

Outros pontos positivos em se utilizar a árvore de decisão estão no fato dela poder ser utilizada em grandes ou pequenos conjuntos de dados, sendo que é possível interpretar modelos pequenos sem a necessidade de utilizar técnicas matemáticas, tornando essa técnica mais eficiente.

Por outro lado, é comum que se tenha a sobreposição dos dados em um modelo, além de problemas na própria modelagem, devido à dependência de divisões paralelas aos ramos. Árvores muito grandes tendem a ficar mais difíceis de se interpretar, visto que certas decisões podem ser consideradas contraditórias. Por fim, é visto que apenas uma mudança nos dados de treinamento pode ocasionar grandes mudanças na lógica de decisão.

2.6.3 *Random Forest*

O método *Random Forest*, ou florestas aleatórias, se baseia na utilização de uma técnica chamada *bagging*. Nela um conjunto de dados, ou *dataset*, é dividido em vários subconjuntos aleatórios de tamanhos equivalentes (BOZKURT et al., 2015).

Através de cada um dos subconjuntos gerados é aplicado o algoritmo de árvore de decisão, apresentando diferentes resultados para um mesmo conjunto global. O resultado final estimado pelo *Random Forest* combina as previsões das árvores de decisão e então seleciona a classe resultante de maior ocorrência (LANTZ, 2015)

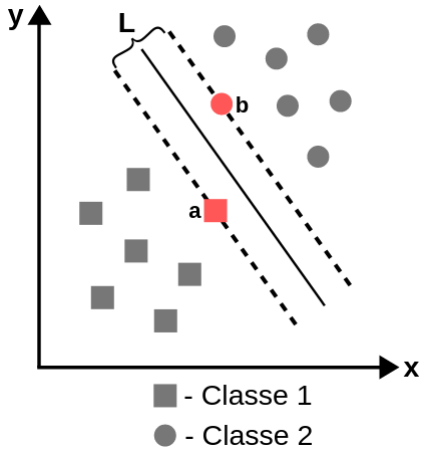
A principal vantagem em se utilizar o algoritmo *Random Forest* está na possibilidade de trabalhar com conjuntos de dados extremamente grandes, visto que na predição são utilizados subconjuntos com muito menos dados. Porém, segundo Lantz (2015), com vários subconjuntos o algoritmo não se torna tão interpretável quanto a árvore de decisão, podendo dificultar o ajuste do modelo às características presentes.

2.6.4 *Support Vector Machine* (SVM)

De acordo com Lantz (2015), o algoritmo *Support Vector Machine* (SVM) pode ser imaginado como uma fronteira que separa as características de cada classe analisada através de um hiperplano.

Em duas dimensões pode-se definir o hiperplano como uma reta que separa classes de forma que ocorra a maximização da sua margem, ou seja, que a distância entre ele e os pontos mais próximos das classes seja a maior possível. Essa maximização é efetiva no caso de classes linearmente separáveis, ou seja, que possam ser corretamente separadas através de uma reta ou plano. A Figura 12 apresenta a maximização da largura L da margem no caso descrito.

Figura 12 – Maximização de um hiperplano em classes linearmente separáveis.



Fonte: Baseado em Lantz (2015).

As linhas tracejadas são normalizados em relação a distância entre um vetor perpendicular ao hiperplano e a origem. Elas são definidas através do produto escalar deste vetor com os vetores de suporte, que partem da origem aos pontos de cada classe mais próximos ao plano (a e b). Segundo Lantz (2015), a largura L da margem pode ser calculada então, através da Equação 2.10, sendo $\vec{a}$ e $\vec{b}$ os vetores de suporte e $\vec{w}$ o vetor perpendicular ao hiperplano.

$$L = \frac{\vec{w} \cdot (\vec{b} - \vec{a})}{\|w\|} = \frac{2}{\|\vec{w}\|} \tag{2.10}$$

Em classes que não são linearmente separáveis, os dados de uma podem sobrepor os das outras. Com isso, realizar a mesma maximização da largura da margem para todos os pontos pode elevar a ocorrência de possíveis *outliers*, prejudicando o processo de classificação. Para corrigir esse problema, uma variável de ajuste pode ser aplicada a todos os pontos de uma classe que violam a largura da margem.

Outra solução possível é mapear as classes de acordo com as funções de *kernel* presentes no SVM. O objetivo é separar as classes de acordo com outros tipos de curvas como a gaussiana, polinomial ou sigmoide (LANTZ, 2015). Com essas funções não lineares, o algoritmo é capaz de construir novas relações matemáticas entre os dados a partir do mapeamento dos mesmos em um número maior de dimensões. O resultado final tende a distinguir tão efetivamente as classes como se fossem linearmente separáveis.

3 DESENVOLVIMENTO

Como visto no Capítulo 2, a comunicação entre dispositivos de uma rede BLE possui certas desvantagens. O uso do parâmetro RSSI na localização destes dispositivos limita a precisão do sistema, uma vez que em ambientes internos a propagação dos sinais é comprometida por conta do multipercurso. Para minimizar esse problema, a coleta dos valores de RSSI dos dispositivos finais por parte dos *gateways* foi realizada no auditório do IFSC-SJ. A escolha deste ambiente se deu por conta do amplo espaço sem obstruções estruturais ou fluxo de pessoas. Os SLAs apresentados na seção 2.6 utilizaram como entrada os arquivos *comma-separated values* (CSV) gerados com os valores de RSSI correspondentes ao mesmo cenário de medição.

Este capítulo descreve na seção 3.1 o *hardware* (*gateways* e dispositivo final) desenvolvido no projeto, enquanto as seções 3.2 e 3.3 apresentam a implementação do servidor e do banco de dados, respectivamente. Por fim, a seção 3.4 apresenta a implementação dos SLAs utilizados nos testes.

3.1 *Hardware*

A comunicação entre os dispositivos BLE ocorre com o dispositivo final enviando seus pacotes de anúncio em *broadcast*. Os *gateways* ficam responsáveis por coletar somente as informações do dispositivo final que possua o *Universally Unique Identifier* (UUID) de serviço específico.

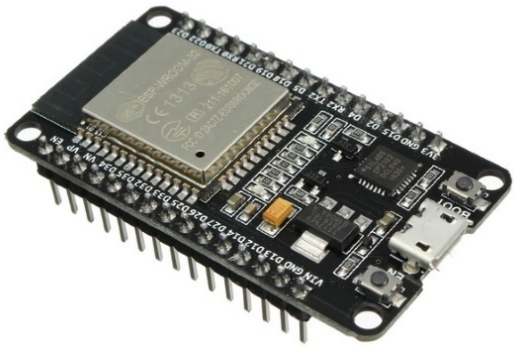
3.1.1 *Gateway*

Para realizar esta função foi escolhida a plataforma de desenvolvimento ESP32[1], mostrada na Figura 13, que possui as interfaces de comunicação sem fio BLE e Wi-Fi (padrões IEEE 802.11b/g/n).

Com essas tecnologias, além da praticidade de implementação por conta de bibliotecas disponíveis para tal, o ESP32 foi uma opção de custo viável para se desenvolver o *gateway*. Ele foi configurado para se comportar como *scanner* BLE, identificando os pacotes de anúncio enviados pelos dispositivos finais e encaminhando os dados da comunicação (RSSI e endereço MAC do dispositivo) via Wi-Fi para o servidor remoto após uma certa quantidade de medidas (determinada no Capítulo 4).

[1] https://www.espressif.com/en/products/hardware/esp32/overview

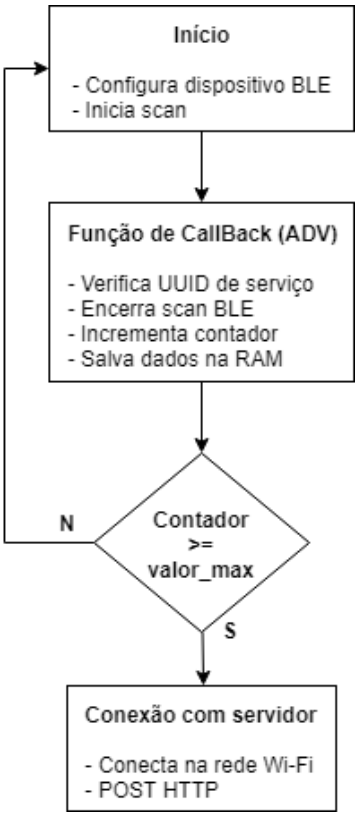
Figura 13 – Plataforma de desenvolvimento ESP32



Fonte: Filipeflop.

O funcionamento dos *gateways* pode ser melhor compreendido através do fluxograma presente na Figura 14.

Figura 14 – Fluxograma de funcionamento dos *gateways*



Fonte: Desenvolvida pelo autor.

O ciclo de execução dos *gateways* começa com a inicialização e configuração do *scanner* BLE. Nela uma função de *callback* é atribuída ao *scanner*, sendo chamada sempre que um novo pacote de anúncio é recebido. Essa função verifica o UUID de serviço do dispositivo encontrado e se este for válido encerra o *scan* BLE.

O processo de salvar os dados na memória RAM consiste na criação de um vetor com a quantidade de posições equivalentes ao número de medidas que se deseja fazer. Sabendo o período em que o dispositivo final envia os pacotes de anúncio, um contador é incrementado de acordo com o tempo que se levou para receber um novo dado, desde o começo de um novo *scan*. Com isso, o RSSI obtido é armazenado na posição equivalente ao valor incrementado do contador. Dessa forma, caso pacotes de anúncio sejam perdidos, o contador é incrementado com o valor equivalente à razão do tempo sem a recepção de um novo dado por parte do *gateway* e o período de *advertising* configurado no dispositivo final. Essa forma de armazenar os dados garante uma espécie de sincronia entre os *gateways*, pois cada posição de seus vetores corresponde ao mesmo sinal transmitido pelo dispositivo final.

Caso o número de medições tenha sido alcançado, os *gateways* iniciam uma conexão Wi-Fi e enviam as informações através de requisições *HyperText Transfer Protocol* (HTTP) do tipo POST. Esse processo é realizado através de um laço de execução, em que a posição do vetor e seu respectivo valor são enviados juntos com os endereços dos *gateways* e do dispositivo final. Caso o número de medidas ainda não tenha sido alcançado, um novo *scanner* BLE é executado e todo ciclo se repete.

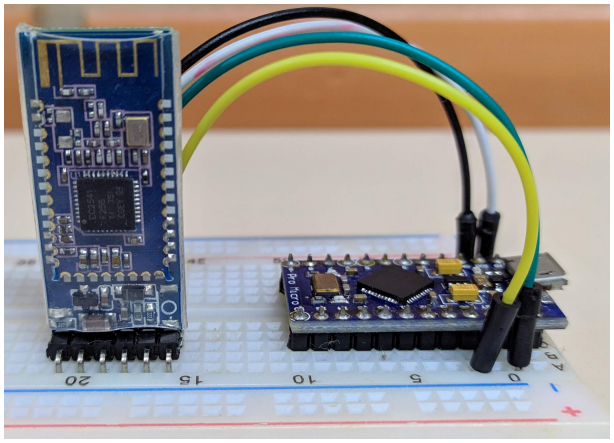
Vale salientar que a plataforma ESP-32 compartilha a mesma antena para ambas as comunicações, por isso não é possível manter a placa conectada a uma rede Wi-Fi durante todo o processo de *scan* do BLE. Além disso, esse modo de execução foi aprimorado desde o início da implementação dos *gateways*. Primeiramente a conexão Wi-Fi e o envio de dados eram realizados a cada pacote de anúncio recebido. Porém, os *gateways* levavam tempos diferentes para se conectar em uma rede, fazendo com que perdessem (ou não) o pacote de anúncio subsequente e consequentemente, a referência dos contadores de medidas. A versão final da implementação garante a identificação correta do RSSI em relação ao mesmo sinal transmitido pelo dispositivo final, dispondo também de uma relação entre o número de pacotes recebidos e perdidos.

3.1.2 Dispositivo final

O dispositivo final envia em um intervalo de tempo pré determinado suas informações (nome, endereço MAC, identificador de serviço, entre outras características) em *broadcast*, através dos pacotes de anúncio do BLE. Para desenvolvê-lo, foi utilizado o módulo BLE AT-09, que utiliza o microcontrolador CC2541[2] da Texas Instruments. Seu papel é enviar constantemente pacotes de anúncio com a sua identificação. Para realizar as configurações no módulo, foi utilizada a plataforma Arduino Pro Micro, estabelecendo uma conexão serial entre as duas plataformas e o computador. A Figura 15 apresenta montagem do protótipo de dispositivo final.

[2] http://www.ti.com/lit/ds/symlink/cc2541.pdf

Figura 15 – Montagem do modulo AT-09 com o Arduino Pro Micro



Fonte: Desenvolvida pelo autor.

A Tabela 1 mostra os comandos de configuração do tipo AT enviados para o AT-09 e seus significados[3]. Após os devidos ajustes, o módulo BLE e o Arduino foram desconectados do computador e alimentados diretamente por uma fonte externa.

Tabela 1 – Comandos AT para configuração do modulo AT-09

Comando	Significado
AT	Confirmação de comunicação
AT+BAUD**4**	Configura taxa de comunicação serial para 9600 bauds
AT+UUID	Verifica UUID de serviço para filtragem no gateway
AT+NAME**BLE1**	Altera nome do dispositivo para "**BLE1**"
AT+ADVI**4**	Configura envio dos dados de anúncio a cada 2 segundos
AT+POWE**2**	Configura potência de transmissão para 0dBm (1 mW)
AT+RESET	reinicia o modulo

Fonte: Desenvolvida pelo autor.

Inicialmente o dispositivo final foi configurado para enviar pacotes de anúncio em intervalos de 4 segundos. Esse tempo foi escolhido para que entre os intervalos de envio, fosse possível analisar cada um dos pacotes recebidos pelos *gateways* via comunicação serial. Após a validação da implementação, a configuração para envio dos pacotes de anúncio a cada 2 segundos foi habilitada e o os trechos do código dos *gateways* destinados à comunicação serial foram comentados.

3.2 Servidor

A integração dos *gateways* com o banco de dados, bem como a geração dos *datasets* em função das medições obtidas, foram realizadas através de um *Web Service* desenvolvido na linguagem *Python*. Sua função é interfacear a comunicação entre diferentes tipos de aplicações, provendo serviços baseados nas operações definidas na sua interface.

[3] http://denethor.wlu.ca/arduino/MLT-BT05-AT-commands-TRANSLATED.pdf

As requisições dos *gateways* são recebidas pelo servidor, responsável por implementar o protocolo HTTP. Os dados externos são representados através de textos do tipo *JavaScript Object Notation* (JSON), que são passados como parâmetro nas requisições do tipo POST para o servidor, sendo tratados de acordo com uma determinada url (*Uniform Resource Locator*). O retorno delas é realizado através dos códigos de resposta do protocolo HTTP, variando-os de acordo com o tratamento da requisição. Um exemplo de requisição do tipo POST pode ser observado no envio de medições para o servidor. Com a criação de um cliente HTTP, o JSON contendo os endereços MAC dos *gateways* e dos dispositivos finais, bem como o RSSI e o contador da medidas, é inserido no cabeçalho da requisição executada através da url: http://192.168.0.13:5001/insere_medicao, sendo o servidor representado com o endereço IP de rede local 192.168.0.13, esperando as requisições na porta 5001.

Com o uso do *Web Service* é possível obter maior confiabilidade nas informações a serem salvas no banco de dados. Quando um novo dado de medição é recebido, por exemplo, verifica-se primeiramente se o sistema está configurado para recebê-lo. Em caso afirmativo, os endereços MAC do *gateway* e do dispositivo final são verificados no banco de dados para que a medida seja finalmente armazenada. Caso o sistema não esteja configurado para receber novas medidas, ou um dos dispositivos não esteja cadastrados no banco de dados, a medida é descartada.

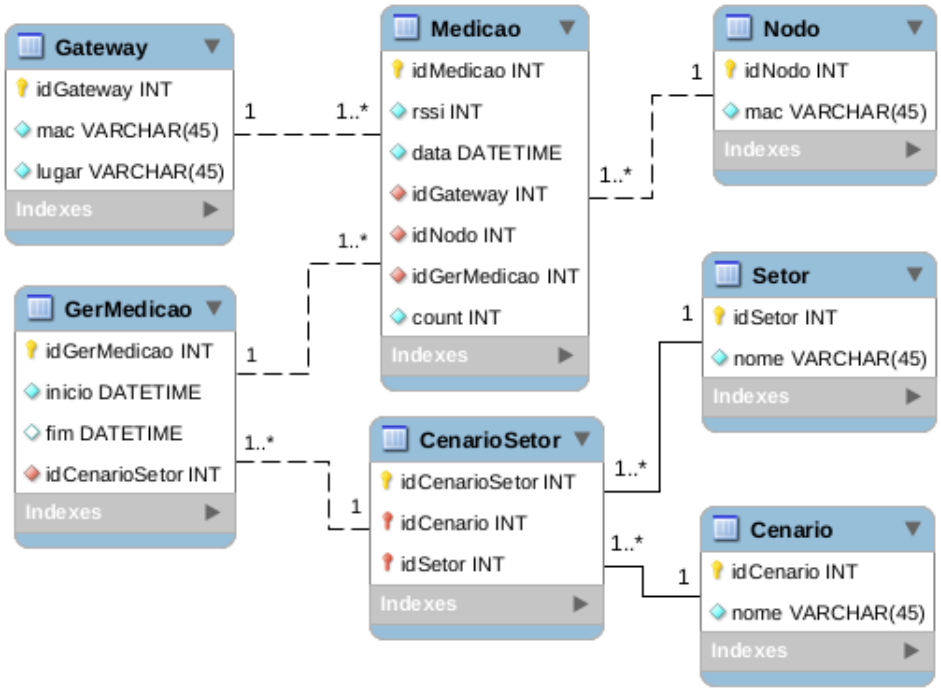
Além deste provedor de serviços, foram criadas aplicações que alimentassem a base de dados do projeto. Funções como adicionar *gateways* e dispositivos finais, bem como criar novos cenários de testes e seus respectivos setores, foram implementadas para tal interação.

Durante o pré projeto deste trabalho, analisou-se a possibilidade de hospedar o servidor e o banco de dados do projeto em um computador remoto disponibilizado pela *Amazon Web Services* (AWS). Porém, a versão gratuita disponível para estudantes correspondia a uma máquina com poder de processamento inferior ao computador utilizado para os testes. Com isso, o processamento da grande quantidade de dados (oriunda dos *datasets* de medições) ficou comprometido com o uso da AWS. Por consequência, optou-se por executar o servidor do projeto na máquina em que os testes já estavam sendo realizados, por ser um computador com maior capacidade de processamento.

3.3 Banco de dados

O banco de dados foi modelado e desenvolvido através da ferramenta MySQL *Workbench*[4], conforme mostra a Figura 16.

Figura 16 – Modelagem do sistema com o MySQL *Workbench*



Fonte: Desenvolvida pelo autor.

A tabela **CenarioSetor** define quantos e quais setores da tabela **Setor** farão parte de um determinado conjunto de medições. O controle desse conjunto é realizado através da tabela **GerMedicao**. Nela, o tempo de início e fim de um conjunto de medições é inserido, de modo que dados que chegaram em um período diferente deste sejam descartados. Além disso, para que não seja necessário modificar o código dos *gateways* a cada mudança de setor, uma nova instância da **GerMedicao** pode ser iniciada para a mesma linha da tabela **Cenário** em que as medidas estão sendo realizadas.

A cada novo conjunto de dados recebido, uma instância da tabela **Medicao** é criada. Nela concentram-se as identificações do *gateway* remetente e do dispositivo final que enviou o pacote de anúncio. A verificação discutida na seção 3.2 atua sobre esses dados de entrada com os previamente armazenados nas tabelas **Gateway** e **Nodo**, sendo que nesta primeira está presente também o lugar em que o *gateway* está posicionado. Além do RSSI, a tabela **Medicao** armazena o valor do contador para a geração dos *datasets* e o horário em que os dados foram salvos.

[4] https://www.mysql.com/products/workbench/

A proposta de salvar as informações no banco de dados entre o intervalo de tempo definido na tabela **GerMedicao** foi desenvolvida em conjunto com a ideia inicial descrita na subseção 3.1.1. Dessa forma, seria possível que os *gateways* ficassem "ativos" o tempo todo, pois os dados só seriam salvos a partir do início de uma nova medição por parte dessa tabela. Mesmo com os *gateways* não se comportando dessa maneria na implementação atual, a funcionalidade foi mantida para controle dos tempos de medição e facilidade na verificação das Medições de cada setor.

3.4 Implementação dos SLAs

Os testes com os algoritmos de aprendizado de máquina supervisionado foram implementados utilizando a linguagem *Python*. Optou-se por utiliza-la também nesta etapa por conta da possível integração com o servidor desenvolvido para o projeto. Foram desenvolvidos algoritmos que estimam a precisão dos classificadores K-NN, *Random Forest* e SVM e em todos eles foram utilizadas as seguintes bibliotecas:

- Pandas[5]: ferramenta para análise dos dados extraídos dos *datasets*;
- *Matplotlib*[6]: geração de gráficos (utilizado também na análise dos dados presentes no Capítulo 4);
- *Scikit-learn*[7]: implementa os algoritmos de aprendizado de máquina.

Como o objetivo deste trabalho não foi desenvolver os algoritmos de aprendizado de máquina, em cada um dos programas de teste os respectivos classificadores eram incluídos através da biblioteca *scikit-learn*.

O fluxo de execução do programa de extração da precisão dos SLAs começa com a inclusão de um arquivo CSV contendo as medições de um determinado cenário de testes. As colunas correspondentes aos valores de RSSI de cada *gateway* são separadas da coluna dos resultados (identificação do setor). Definiu-se que 80% dos dados destes conjuntos seriam separados para a etapa de treinamento dos classificadores, destinando o restante à etapa de testes. Essa separação é realizada através da função *train_test_split()* da biblioteca *scikit-learn*, que distribui os *datasets* de forma aleatória. Com isso, para verificar a razão entre os dados estimados de forma correta e todo o conjunto, um laço de execução foi realizado, extraindo-se a média das precisões de cada algoritmo.

[5] https://pandas.pydata.org/
[6] https://matplotlib.org/
[7] https://scikit-learn.org/stable/

Antes de estimar a precisões de todos os algoritmos, o classificador K-NN verifica a quantidade de vizinhos próximos que oferecem a melhor precisão. Esse processo é realizado através de outro laço de execução interno ao primeiro, variando o valor de K. De forma análoga, o classificador *Random Forest* varia a quantidade de árvores de decisão utilizadas, buscando salvar os dados referentes à quantidade de árvores que resultaram em uma melhor precisão. Após a conclusão dos dois laços de execução (laço externo para criação dos *datasets* de treinamento e teste e laço interno para variação dos parâmetros dos SLAs), a média das precisões obtidas com os determinados parâmetros de cada algoritmo é extraída.

No caso do SVM, a cada novo *dataset* de treinamento e teste, foram salvos os dados preliminares de acordo com as funções de *kernel* linear e gaussiana. Por fim, a média das precisões com cada uma dessas funções foi realizada para estimar o resultado final do classificador.

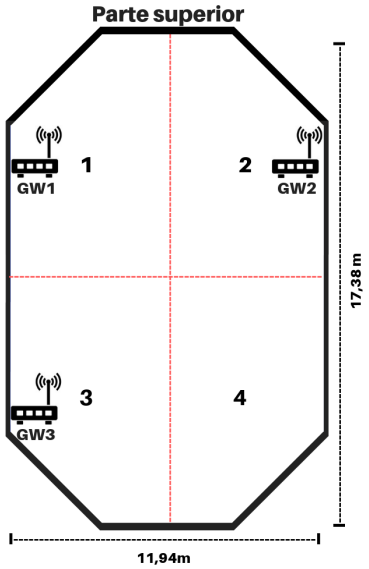
4 TESTES E RESULTADOS

Os testes realizados no auditório do IFSC-SJ consistiram em dividi-lo em 4 e 24 setores. Durante todos os experimentos não houve fluxo de pessoas pelo ambiente e os *gateways* foram mantidos nas mesmas posições, com a mesma altura em relação ao chão (90 centímetros). O dispositivo final foi posicionado sempre no centro de cada setor, além de ter a potência de transmissão fixada em 0dBm (1 mW) e o envio de pacotes de anúncio a cada 2 segundos.

4.1 Segmentação do ambiente em 4 setores

Inicialmente, o auditório foi dividido em 4 setores de mesmo tamanho, realizando as coletas com o dispositivo final posicionado no centro de cada setor, conforme apresentado na Figura 17, a uma altura de 80 centímetros. Nesta situação, as dimensões da cada setor foram aproximadamente 5,97 m de largura por 8,68 m de comprimento.

Figura 17 – Posicionamento dos *gateways* no auditório

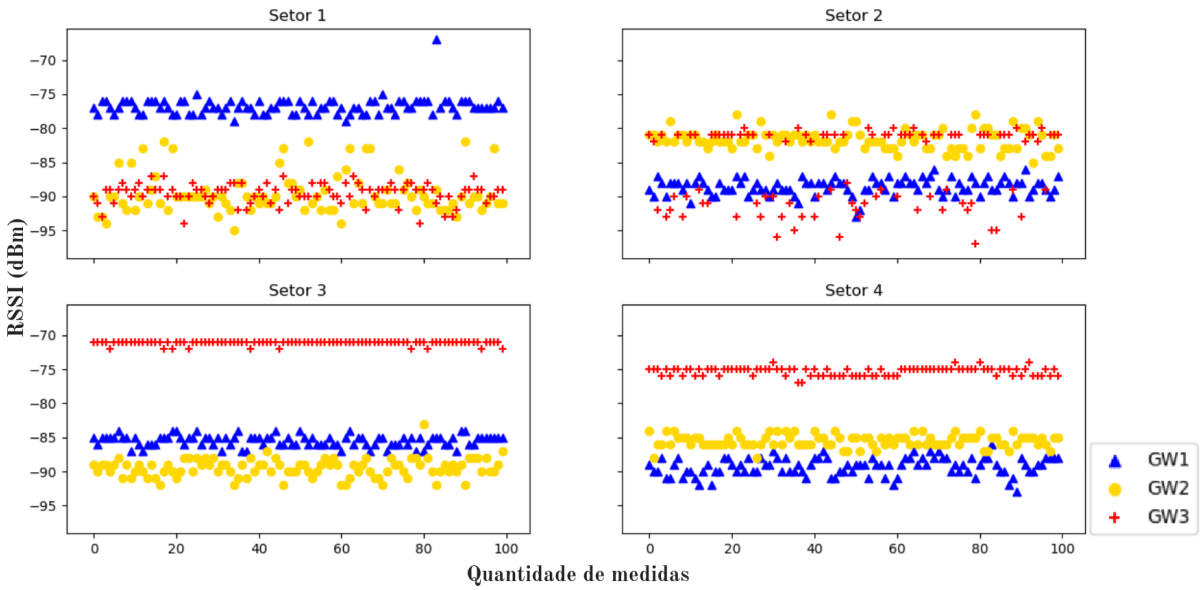


Fonte: Desenvolvida pelo autor.

Os experimentos iniciais consideraram 900 medidas de RSSI (300 de cada *gateway*) por setor. No entanto, foi observado que mesmo com a grande quantidade de pontos, a distribuição dos dados coletados em cada *gateway* apresentava poucas dispersões em torno do valor médio. Com isso, optou-se por diminuir o *dataset* para 300 amostras de cada setor, minimizando o processamento do computador durante a manipulação dos conjuntos de dados e a aplicação dos SLAs.

A Figura 18 apresenta os gráficos de dispersão dos valores de RSSI obtidos em cada um dos setores. Através dessa representação é possível observar a concentração dos sinais recebidos por cada *gateway* em diferentes intensidades de potência (eixo das ordenas), de acordo com a quantidade de medições realizadas (eixo das abscissas).

Figura 18 – Gráfico de dispersão do primeiro experimento

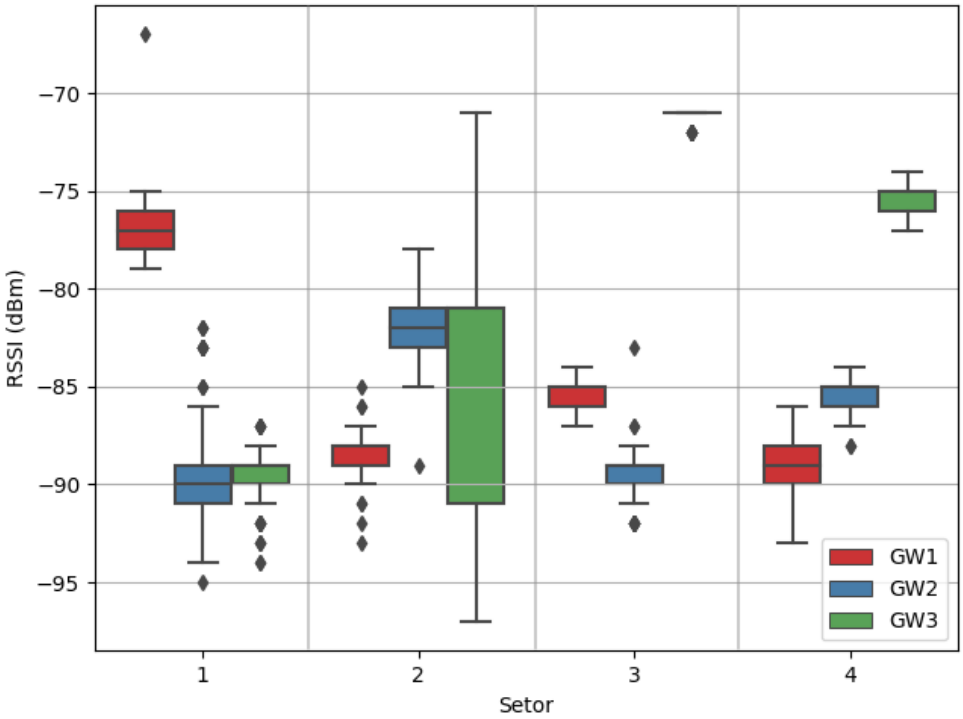


Fonte: Desenvolvida pelo autor.

Os três primeiros setores estavam mais próximos dos *gateways* 1, 2 e 3, respectivamente. Como era de se esperar, em cada um deles foram verificados valores de RSSI maiores em relação ao *gateway* em questão. As amostras mais discrepantes relacionadas ao *gateway* 2 no setor 1 e ao *gateway* 3 no setor 2 podem estar associadas à altura em que o dispositivo final foi posicionado. Por estar praticamente à mesma altura das cadeiras do auditório, os sinais propagados por ele podem ter sofrido atenuações causadas por tais obstruções. Nos últimos dois setores o dispositivo final ficou posicionado no palco, onde não possuía obstruções próximas, favorecendo a propagação dos sinais transmitidos e obtendo menores variações do RSSI em cada *gateway*.

Uma forma de se obter dados mais detalhados sobre cada setor é através dos diagramas de caixa presentes na Figura 19. Através deles é possível extrair os valores máximos, mínimos e a média da potência do sinal recebido por cada *gateway*. Além disso, possível verificar a presença dos *outliers*, ou valores discrepantes, que correspondem às medidas fora dos limites estipulados pelos quartis de cada conjunto de medidas.

Figura 19 – Diagrama de caixa das amostras coletadas no primeiro experimento



Fonte: Desenvolvida pelo autor.

Nota-se que mesmo com uma pequena quantidade de pontos, o agrupamento das medidas em cada setor é bem delimitado. Esse comportamento pode ser justificado pela mudança quase nula do ambiente durante os períodos de coleta. Com isso, todos os SLAs obtiveram precisões muito próximas ou iguais a 100% na estimativa dos setores em que o dispositivo final esteve, mesmo sem desconsiderar os *outliers*. No caso do K-NN, a quantidade de vizinhos próximos utilizados na classificação não influenciou na precisão. No algoritmo *Random Forest*, houve variação somente com o uso de uma única árvore de decisão, resultando em uma precisão de 98,75%.

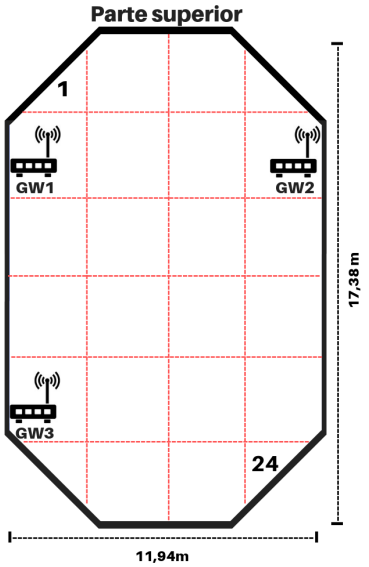
Por mais que não tenha sido possível analisar com efetividade qual algoritmo apresentou o melhor resultado, este experimento foi útil para analisar a distribuição dos valores de RSSI obtidos por cada *gateway* e entender e utilizar as bibliotecas do *Python* destacadas na seção 3.4. Verificou-se que uma quantidade pequena de amostras foi suficiente para estimar com total precisão a localização do dispositivo final. Além disso, notou-se que a altura em que o dispositivo final foi fixado pode ter prejudicado as medições, pois por ter pouca diferença em relação às cadeiras do auditório, muitos sinais eram atenuados ao ponto de não serem identificados pelos *gateways* mais distantes.

4.2 Segmentação do ambiente em 24 setores

Este experimento teve como objetivo verificar a precisão dos classificadores com o auditório dividido em um número maior de setores. Primeiramente, o conjunto dos dados obtidos foi utilizado diretamente pelos SLAs, analisando os parâmetros que resultassem nas melhores precisões. Na segunda etapa do experimento, o *dataset* original foi modificado, de forma a desconsiderar os valores mais discrepantes, ou *outliers* obtidos.

O auditório foi segmentado em 24 setores (6 linhas por 4 colunas) de 2,985 m de largura por 2,89 m comprimento, conforme mostra a Figura 20.

Figura 20 – Divisão do auditório no segundo experimento

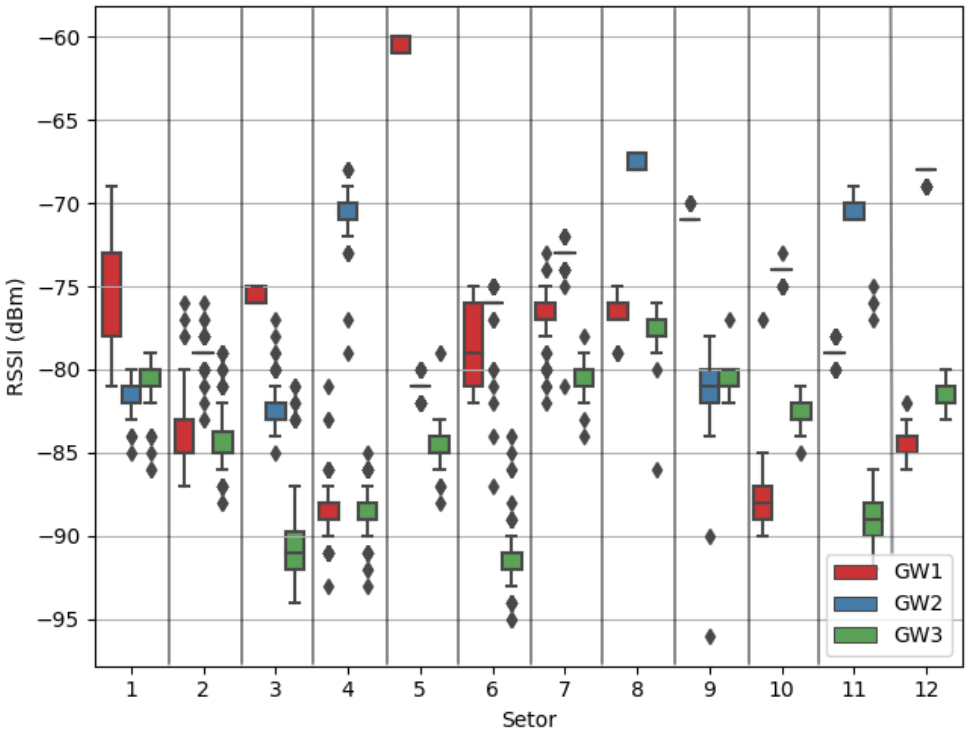


Fonte: Desenvolvida pelo autor.

Em contraste com o primeiro experimento, o dispositivo final foi fixado a uma altura de 1,6 m, a fim de se verificar possíveis melhorias relacionadas à quantidade de pacotes de anúncio, bem como o nível de potência em que estes eram recebidos por cada *gateway*.

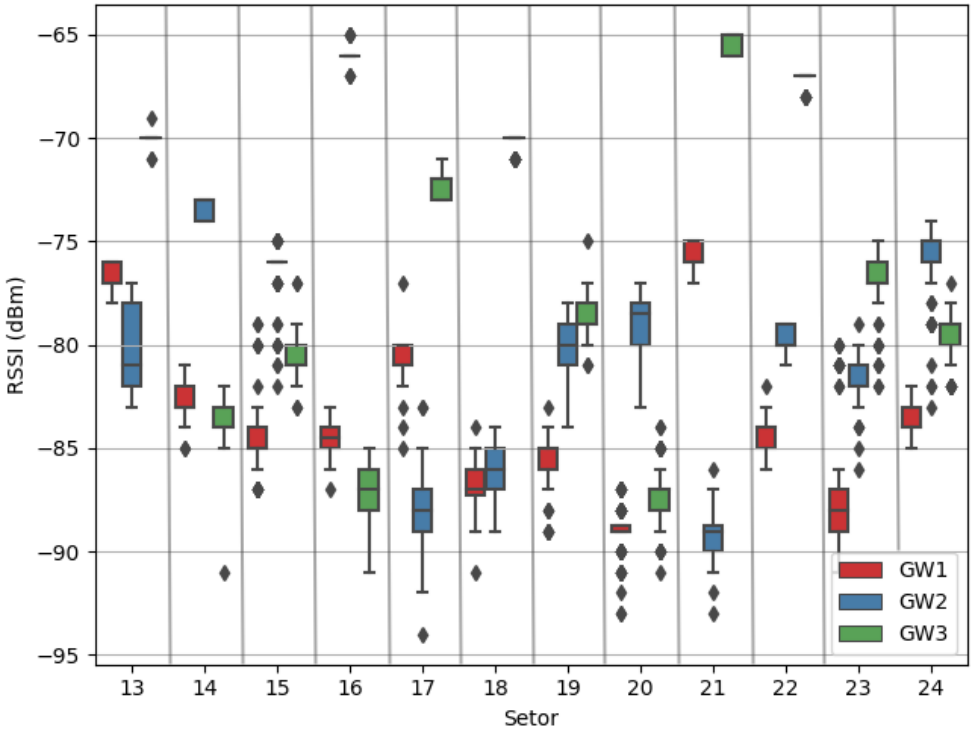
O *dataset* gerado nesta etapa agrupou 100 amostras de cada setor, no qual suas estatísticas podem ser observadas através dos diagramas de caixa presentes nas figuras 21 e 22.

Figura 21 – Diagramas de caixa das amostras coletadas nos 12 primeiros setores do segundo experimento



Fonte: Desenvolvida pelo autor.

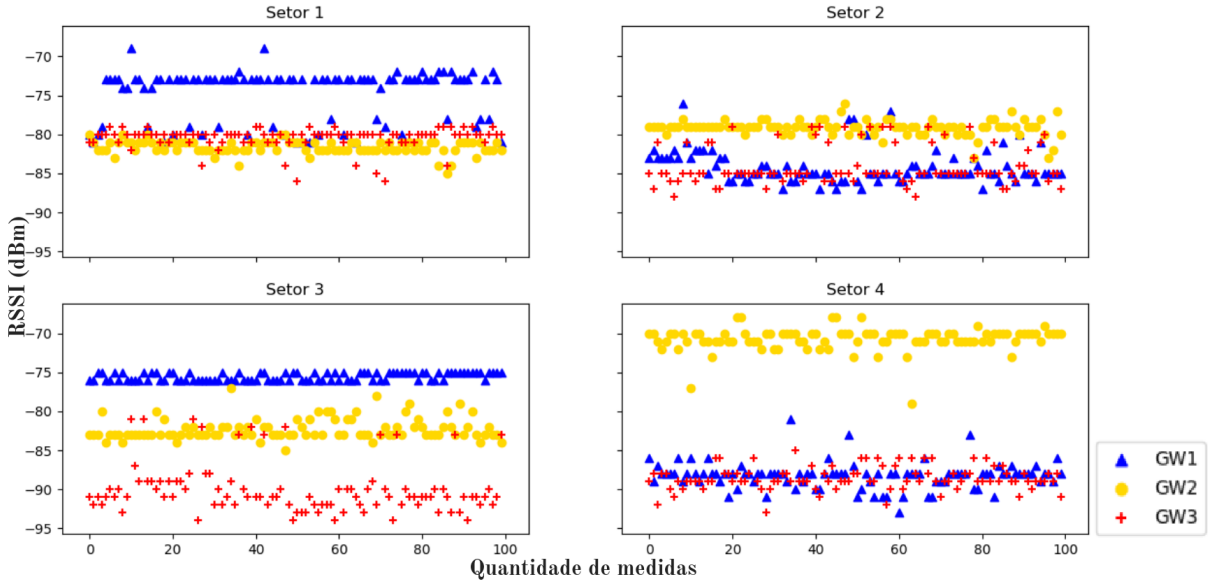
Figura 22 – Diagramas de caixa das amostras coletadas nos 12 últimos setores do segundo experimento



Fonte: Desenvolvida pelo autor.

Nota-se que mesmo com o auditório dividido em mais segmentos, os valores de RSSI predominantes dos *gateways* em cada setor podem ser visualmente distinguidos. O desvio padrão elevado e a ocorrência de muitos *outliers* nos setores 1, 2, 3 e 4 podem ser justificados pelo fato deles estarem localizados embaixo da marquise do auditório (parte superior) e próximos à parede, dificultando a propagação do sinal transmitido pelo dispositivo final. A Figura 23 apresenta a dispersão das amostras obtidas pelos três *gateways* nestes setores.

Figura 23 – Gráfico de dispersão dos 4 primeiros setores do segundo experimento

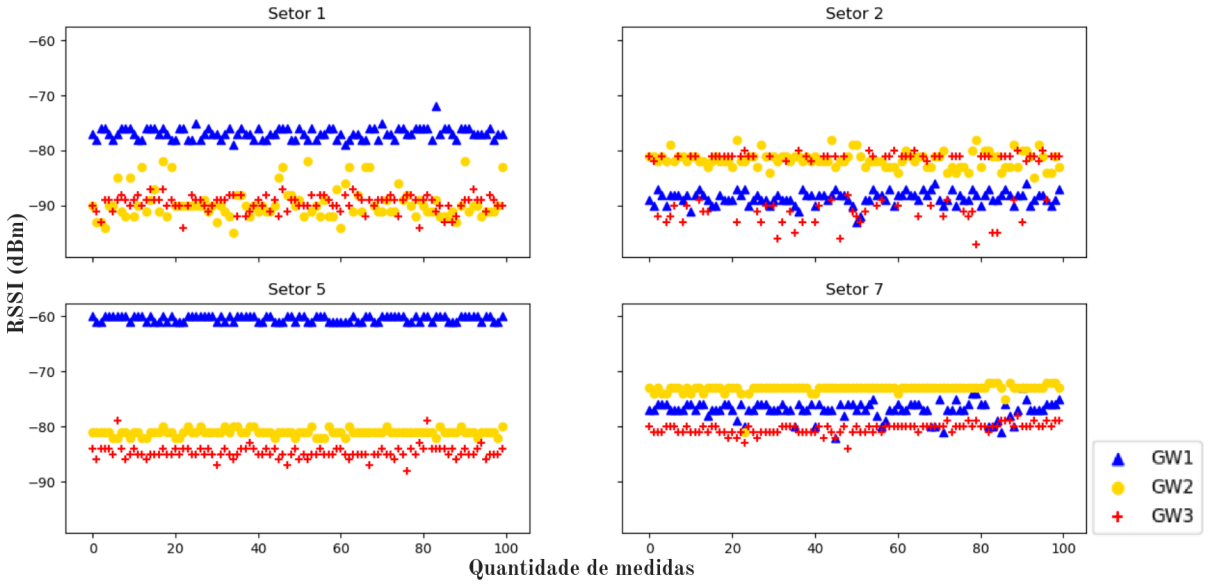


Fonte: Desenvolvida pelo autor.

Verificou-se também que a fixação do dispositivo final em uma altura maior que a do primeiro experimento resultou em amostras mais precisas nos setores localizados entre as cadeiras do auditório. A Figura 24 apresenta uma comparação entre os setores 1 e 2 do primeiro experimento, e os setores 5 e 7 do segundo (mais próximos fisicamente do centro dos setores do primeiro experimento). A intensidade da potência do sinal recebido por cada *gateway* também foi maior, uma vez que os sinais não sofreram tanta atenuação por conta das obstruções do primeiro experimento.

A primeira parte deste experimento manteve os *outliers* do conjunto de medições coletadas. Para verificar a precisão dos classificadores, foram utilizados os mesmos *datasets* de treinamento e teste extraídos dele. Como relatado na seção 3.4, os *datasets* foram divididos de forma pseudo aleatória através da função *train_test_split()* presente na biblioteca *Scikit-learn*. Pelo fato das amostras de cada setor não serem iguais umas às outras, o uso de diferentes valores de sementes para a separação aleatória dos *datasets* fazia com que os dados de treinamento e teste variassem, alterando consequentemente a precisão dos classificadores. Porém, como a função *train_test_split()* não exige que se tenha um valor pré definido de semente, caso este não seja passado como parâmetro, um

Figura 24 – Comparação entre a dispersão dos dois primeiros experimentos



Fonte: Desenvolvida pelo autor.

número aleatório é gerado automaticamente e utilizado como tal. Com isso, a média da precisão dos classificadores foi calculada para 100 *datasets* de treinamento e teste extraídos de foram aleatória. Os resultados podem ser observados na Tabela 2.

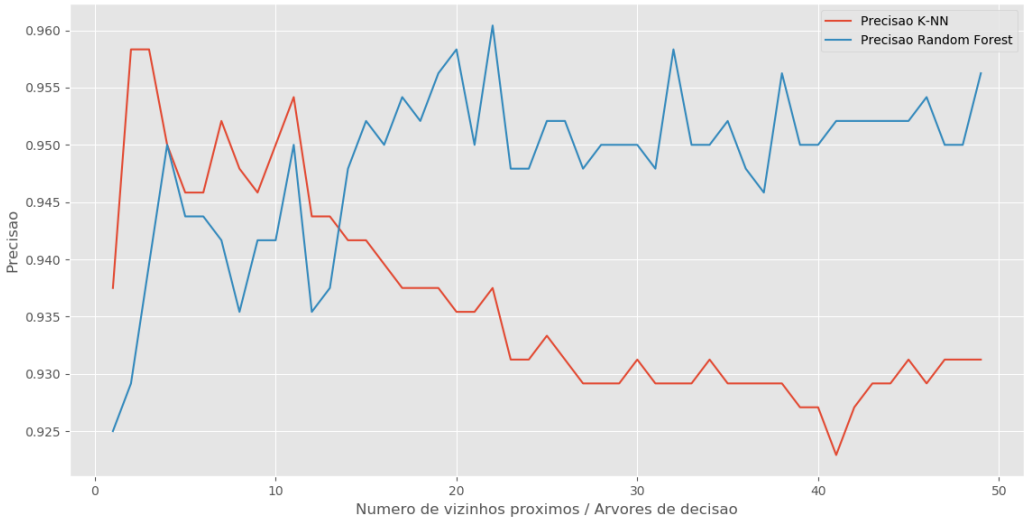
Tabela 2 – Média das precisões obtidas no segundo experimento

Classificador	Precisão
K-NN	95,61%
Random Forest	96,15%
SVM (linear)	94,17%
SVM (gaussiana)	94,99%

Fonte: Desenvolvida pelo autor.

Através do laço de execução realizado para obter os valores da Tabela 2, também foi analisada a quantidade de vizinhos próximos utilizados no algoritmo K-NN e de árvores de decisão do *Random Forest*, que ofereceram os melhores resultados. A Figura 25 apresenta a precisão destes dois algoritmos de acordo com a variação de seus respectivos parâmetros. Nota-se a relação inversamente proporcional da precisão obtida em cada classificador, de acordo com o aumento do quantidade de vizinhos próximos e árvores de decisão utilizados. Apesar da variação, o K-NN teve uma precisão maior utilizando em média 3 vizinhos próximos, enquanto o *Random Forest* classificou os setores com maior precisão utilizando uma média de 23 árvores de decisão.

Figura 25 – Precisão do Classificadores K-NN e *Random Forest* variando seus parâmetros



Fonte: Desenvolvida pelo autor.

A segunda etapa do experimento consistiu em retirar os *outliers* identificados nos diagramas de caixa das figuras 21 e 22. Os SLAs foram aplicados ao *dataset* modificado, obtendo as precisões apresentadas na Tabela 3.

Tabela 3 – Média das precisões obtidas com a exclusão dos *outliers* do *dataset*

Classificador	Precisão
K-NN	99,17%
Random Forest	99,15%
SVM (linear)	98,81%
SVM (gaussiana)	98,95%

Fonte: Desenvolvida pelo autor.

Foi verificada nessa análise uma melhora significativa na precisão de todos os algoritmos, chegando a quase 100% com o K-NN e o *Random Forest*. Isso se deu pelo fato das amostras de cada setor estarem distribuídas de uma forma mais homogênea. Como relatado na subseção 2.6.4, o algoritmo SVM atua sobre a formação de um hiperplano linear (por padrão) para distinguir os dados de cada classe. Logo, a presença dos *outliers* podia causar uma certa limitação na maximização do hiperplano, diminuindo assim, a precisão do classificador. Com o algoritmo *Random Forest*, a comparação dos valores de RSSI dos *gateways* em cada setor é realizada através de múltiplas árvores de decisão. Caso essas árvores utilizassem um valor discrepante nos pontos de decisão, a escolha errada de um ramo também influenciaria negativamente na classificação. O algoritmo K-NN também teve seus resultados influenciados com a remoção dos *outliers*. Quando considerados na primeira parte, esses valores poderiam ser correspondidos aos vizinhos mais próximos de uma determinada amostra, fazendo com que o algoritmo compreendesse erroneamente que esta pertencesse à classe do *outlier*. A homogeneidade das classes fez com que a

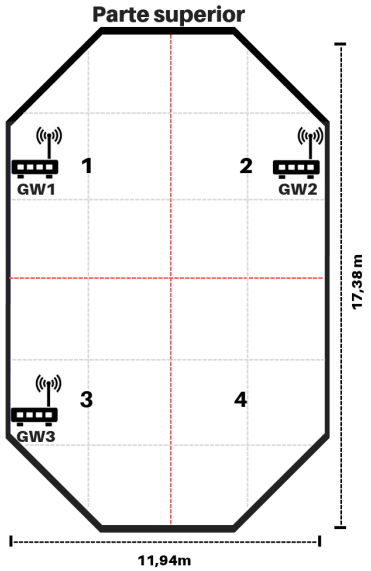
classificação das amostras de teste ocorresse de forma mais justa, uma vez que os valores mais discrepantes não estariam influenciando nas ações do algoritmo.

Em relação aos parâmetros utilizados pelos algoritmos K-NN e *Random Forest*, observou-se que em média, a maior precisão do K-NN era obtida comparando as amostras de teste com 6 vizinhos próximos. Já no com o *Random Forest*, foram utilizadas em média 9 árvores de decisão para se obter a precisão apresentada na Tabela 3. Como os parâmetros da função de *kernel* gaussiana do classificador SVM são reconfigurados a cada amostra do *dataset*, sua distribuição consegue sustentar com maior precisão a variação dos valores de RSSI. Com isso, foi possível obter uma precisão média ligeiramente superior à aplicação do mesmo algoritmo com a função de *kernel* linear.

4.3 Segmentação do ambiente em 4 macro setores

O objetivo deste experimento foi verificar a precisão dos classificadores através do mesmo conjunto de medições obtidas na seção 4.2, porém agrupando os 24 setores originais em 4 setores de maior área, denominados macro setores, conforme mostra a Figura 26. Cada macro setor dispõe das mesmas dimensões do experimento relatado na seção 4.1.

Figura 26 – Formação dos macro setores através do *dataset* do segundo experimento

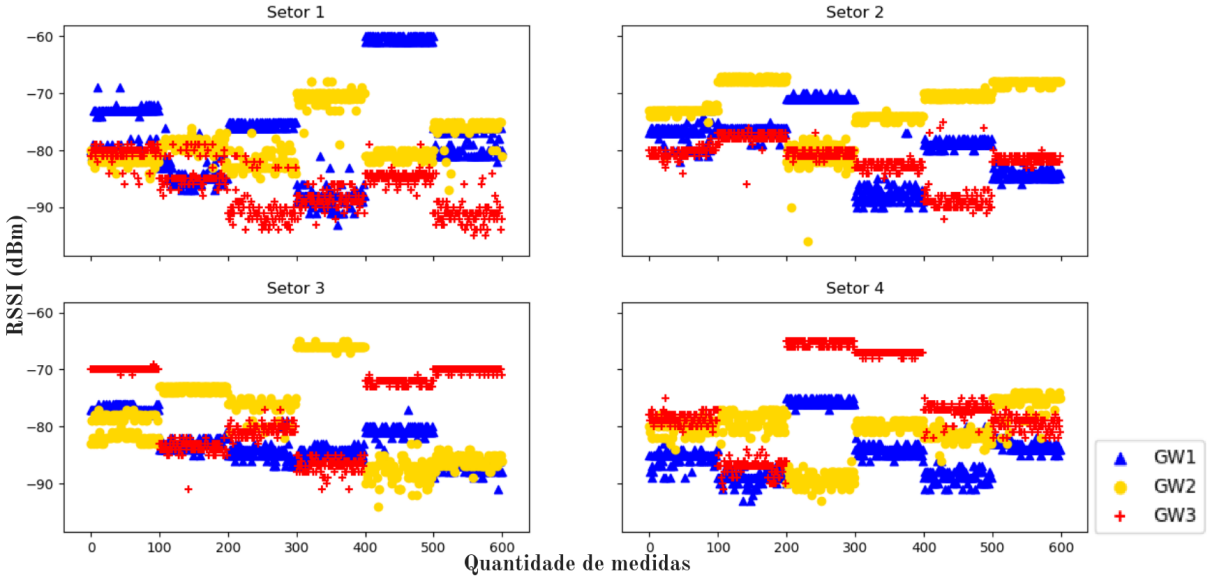


Fonte: Desenvolvida pelo autor.

Em contraste com as medições anteriores, em que as amostras de RSSI eram coletadas com o dispositivo final fixado no centro de cada setor do auditório, este experimento apresenta 6 conjuntos de dados correspondentes a diferentes posições de um mesmo macro setor. Como cada um destes possui suas próprias amostras, é possível ter um conhecimento mais detalhado sobre a distribuição dos dados pertencentes ao mesmo agrupamento. Isso possibilitou definir os limitantes de cada macro setor de forma mais especifica, garantindo

classificações mais confiáveis. A Figura 27 apresenta os gráficos de dispersão de cada macro setor, de acordo com as amostras coletadas por cada *gateway*.

Figura 27 – Gráfico de dispersão dos macro setores



Fonte: Desenvolvida pelo autor.

Para extrair a precisão dos SLAs, foi realizado o mesmo processo exposto na seção 4.2, com a presença dos *outliers* no *dataset*. A média da precisão de cada classificador pode ser observada na Tabela 4.

Tabela 4 – Média das precisões obtidas no terceiro experimento

Classificador	Precisão
K-NN	97,48%
Random Forest	97,61%
SVM (linear)	64,97%
SVM (gaussiana)	96,81%

Fonte: Desenvolvida pelo autor.

Nota-se uma estimativa mais precisa do algoritmo SVM utilizando a função de *kernel* gaussiana em relação à linear. A melhoria com o uso dessa função pode ser justificada pelo fato dos *datasets* de cada setor não estarem mais agrupados de forma tão concisa como nos outros experimentos. A inclusão de diferentes conjuntos de dados que representam um mesmo macro setor, define relações distintas entre todas as amostras, o que não acontecia nos outros experimentos, pois só se tinha um conjunto de dados por setor. Essa característica fez com que o uso do classificador SVM com a função de *kernel* linear não possuísse uma precisão tão boa quanto a gaussiana.

Os classificadores K-NN e *Random Forest* possuíram resultados próximos, tendo um desempenho ligeiramente maior que o SVM utilizando a função de *kernel* gaussiana. Os valores médios de vizinhos próximos que resultaram nas melhores classificações do K-NN foram de 8, enquanto foram utilizadas em média 18 árvores de decisão para se obter a melhor precisão do algoritmo *Random Forest*.

Removendo os *outliers* do conjunto de dados, os classificadores apresentaram resultados semelhantes à segunda etapa do experimento anterior, com exceção da utilização da função de *kernel* linear no algoritmo SVM, que como já relatado, apresentou a pior precisão. A Tabela 5 exibe as médias das precisões obtidas em cada algoritmo. A média de vizinhos próximos do K-NN e de árvores de decisão do algoritmos *Random Forest* que resultaram nas melhores precisões foram de 4 e 7, respectivamente.

Tabela 5 – Média das precisões obtidas no terceiro experimento com a exclusão dos *outliers*

Classificador	Precisão
K-NN	99,33%
Random Forest	99,35%
SVM (linear)	66,66%
SVM (gaussiana)	99,14%

Fonte: Desenvolvida pelo autor.

5 CONCLUSÕES

Este trabalho teve como objetivo analisar diferentes algoritmos de aprendizado de máquina supervisionado (SLAs) aplicadas à localização *indoor*. Foram utilizados os classificadores *K-Nearest Neighbor* (K-NN), *Random Forest* e *Support Vector Machine* (SVM), variando seus parâmetros para obter a melhor precisão possível em cada cenário de teste. Conforme mostram os resultados obtidos, os SLAs conseguiram resultados satisfatórios na classificação dos setores em que o auditório do IFSC-SJ foi dividido, baseando-se apenas no parâmetro RSSI.

Pelo fato de não haver obstruções estruturais e fluxo de pessoas no ambiente em que os testes foram realizados, as amostras de RSSI não sofreram grandes variações. Essa estabilidade favoreceu a coleta de uma quantidade reduzida de medidas, totalizando 100 amostras por *gateway* em cada setor.

Os classificadores K-NN e *Random Forest* obtiveram resultados muito próximos em todos os testes, superando levemente o SVM utilizando as funções de *kernel* linear e gaussiana. A inferioridade do SVM em relação aos demais, neste caso, é acentuada pelo fato dele demandar mais processamento durante a constante atualização do classificador.

A variação do segundo experimento mostrou que para dados extremamente concisos é possível obter precisões muito próximas de 100% com todos os classificadores, mesmo com uma quantidade elevada de setores. A retirada dos valores discrepantes dos *datasets* foi suficiente neste tipo de cenário, porém, em cenários reais (com fluxo de pessoas e obstruções físicas) a média dos valores de RSSI coletados tende a variar mais. Sendo assim, a simples exclusão dos *outliers* pode não ser suficiente para estimar a localização do dispositivo final com tanta precisão.

Foi verificado que para um conjunto de dados que não são linearmente separáveis, a utilização da função de *kernel* linear do algoritmo SVM obteve resultados muito inferiores se comparada aos demais classificadores e à utilização do próprio SVM com a função de *kernel* gaussiana. Isso pode ter ocorrido por conta da maximização do hiperplano linear não conseguir separar efetivamente todas as amostras de um mesmo setor.

Por mais que tenham sido coletadas poucas amostras de RSSI em cada *gateway*, a comunicação entre os dispositivos não se demonstrou muito efetiva no primeiro experimento. Como era necessário que o mesmo sinal transmitido pelo dispositivo final fosse identificado pelos três *gateways*, a não identificação de um pacote de anúncio por apenas um deles já o invalidava. Com isso e aliado qualidade das antenas impressas utilizadas nas plataformas ESP-32 e no módulo BLE AT-09, a perda de pacotes era constante, fazendo com que a coleta de apenas 100 amostras válidas demorasse mais do que o previsto.

Para minimizar esse problema, o dispositivo final foi colocado a uma altura maior no segundo experimento, desobstruindo a linha de visada com os *gateways*. Através desta modificação, observou-se que mais pacotes de anúncio foram identificados por eles e que os valores de RSSI obtidos apresentaram uma variação menor se comparados ao primeiro experimento. Essa melhora ocorreu possivelmente devido à diminuição da reflexão e atenuação dos sinais transmitidos pelo dispositivo final.

5.1 Trabalhos futuros

Como relatado, este trabalho focou em realizar a localização *indoor* em um único ambiente, limitando-se a utilizar o mínimo de dispositivos de acordo com a técnica de localização *fingerprinting*. Partindo da mesma técnica, estudos mais profundos podem ser realizados a respeito da aplicação do *Bluetooth Low Energy* (BLE) em ambientes com repartições ou fluxo de pessoas, como escritórios ou laboratórios. Uma quantidade maior de dispositivos possivelmente seria necessária para validar os experimentos. Além disso, outras abordagens estatísticas relacionadas à manipulação dos *datasets* teriam de ser estudadas, para que se pudesse obter com maior fidelidade a precisão de cada algoritmo.

Com a disposição do código fonte do servidor e o esquema do banco de dados, é possível implementar uma solução totalmente integrada em que se possa validar a localização de dispositivos através do desenvolvimento de uma aplicação específica.

Através dos *datasets* gerados nos experimentos, outros algoritmos de aprendizado de máquina podem ser implementados, realizando comparações com os analisados neste trabalho, a fim de estudar a eficiência (precisão por tempo de processamento) de cada um deles.

Com o objetivo de suprir com as limitações do BLE, outra possível solução para minimizar a perda de dados por parte dos *gateways*, seria configurar o dispositivo final para enviar uma rajada de pacotes de anúncio em um intervalo de tempo muito pequeno, havendo assim, pouca variação do meio. Com essa redundância, a chance dos *gateways* não identificarem um dos sinais enviados em rajada seria menor. Alternativamente, um objeto de pesquisa interessante seria realizar a localização *indoor* utilizando outras tecnologias de comunicação que não sofram tanta atenuação por obstruções estruturais, explorando assim, a localização *indoor* em diferentes ambientes.

Por fim, os parâmetros da comunicação entre os dispositivos podem ser combinados com outros tipos de informações, como dados de sensores. Dependendo do cenário e do tipo de sensor, variações predominantes em setores específicos podem resultar em uma estimativa mais precisa por parte dos SLAs.

REFERÊNCIAS

AL-FUQAHA, A. et al. Internet of things: A survey on enabling technologies, protocols, and applications. *Communications Surveys & Tutorials, IEEE*, IEEE, USA, v. 17, n. 4, p. 2347–2376, 2015. ISSN 1553-877X. Citado 3 vezes nas páginas 21, 25 e 26.

ATHMAJA, S.; HANUMANTHAPPA, M.; KAVITHA, V. A survey of machine learning algorithms for big data analytics. In: . Coimbatore, Tamil Nadu, India: [s.n.], 2018. v. 2018-January, p. 1 – 4. Disponível em: <http://dx.doi.org/10.1109/ICIIECS.2017.8276028>. Acesso em: 20 nov. 2018. Citado na página 40.

BOZKURT, S. et al. A comparative study on machine learning algorithms for indoor positioning. In: . Madrid, Spain: [s.n.], 2015. Disponível em: <http://dx.doi.org/10.1109/INISTA.2015.7276725>. Citado na página 42.

CHANG, K.-H. Bluetooth: A viable solution for iot? *Wireless Communications*, IEEE, USA, p. 6–7, 2014. Citado 3 vezes nas páginas 35, 37 e 39.

CUI, L. et al. A survey on application of machine learning for internet of things. *International Journal of Machine Learning and Cybernetics*, Springer Berlin Heidelberg, Berlin/Heidelberg, v. 9, n. 8, p. 1399–1417, 2018. ISSN 1868-8071. Citado na página 39.

DARROUDI, S. M.; GOMEZ, C. Bluetooth low energy mesh networks: A survey. *Sensors (Switzerland)*, v. 17, n. 7, 2017. ISSN 14248220. Citado na página 35.

FARID, Z.; NORDIN, R.; ISMAIL, M. Recent advances in wireless indoor localization techniques and system. *Journal of Computer Networks and Communications*, Hindawi Publishing Corporation, v. 2013, 2013. ISSN 2090-7141. Citado 3 vezes nas páginas 29, 33 e 34.

GOMEZ, C.; OLLER, J.; PARADELLS, J. Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. *Sensors (Basel, Switzerland)*, Molecular Diversity Preservation International (MDPI), v. 12, n. 9, p. 11734–11753, 2012. ISSN 1424-8220. Citado 4 vezes nas páginas 35, 36, 37 e 38.

GU, Y.; LO, A.; NIEMEGEERS, I. A survey of indoor positioning systems for wireless personal networks. *Communications Surveys & Tutorials, IEEE*, IEEE, USA, v. 11, n. 1, p. 13–32, 2009. ISSN 1553-877X. Citado 2 vezes nas páginas 29 e 34.

LANTZ, B. *Machine Learning with R*. 2nd. ed. [S.l.]: Packt Publishing, 2015. ISBN 1784393908, 9781784393908. Citado 4 vezes nas páginas 40, 41, 42 e 43.

LIU, H. H. et al. Survey of wireless indoor positioning techniques and systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, IEEE, USA, v. 37, n. 6, p. 1067–1080, 2007. ISSN 1094-6977. Citado 3 vezes nas páginas 22, 26 e 27.

LOUREIRO, A. A. et al. Redes de sensores sem fio. In: SN. *Simpósio Brasileiro de Redes de Computadores (SBRC)*. [S.l.], 2003. p. 179–226. Citado na página 25.

LOVON-MELGAREJO, J. et al. Supervised learning algorithms for indoor localization fingerprinting using ble4.0 beacons. In: . Arequipa, Peru: [s.n.], 2018. v. 2017-November, p. 1 – 6. Disponível em: <http://dx.doi.org/10.1109/LA-CCI.2017.8285716>. Acesso em: 10 nov. 2018. Citado na página 40.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. *Sistemas Inteligentes-Fundamentos e Aplicações*, v. 1, n. 1, p. 32, 2003. Citado na página 39.

PU, Y.-C.; YOU, P.-C. Indoor positioning system based on ble location fingerprinting with classification approach. *Applied Mathematical Modelling*, v. 62, p. 654 – 663, 2018. ISSN 0307-904X. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0307904X18302841>. Acesso em: 25 out. 2018. Citado 2 vezes nas páginas 33 e 36.

QIU, J. et al. A survey of machine learning for big data processing. *EURASIP Journal on Advances in Signal Processing*, v. 2016, n. 1, p. 67, May 2016. ISSN 1687-6180. Disponível em: <https://doi.org/10.1186/s13634-016-0355-x>. Acesso em: 22 nov. 2018. Citado na página 39.

RAPPAPORT, T. *Wireless Communications: Principles and Practice, 2nd Edition*. [S.l.: s.n.], 2001. v. 2. ISBN 0130422320. Citado na página 27.

REZENDE, E. M.; YNOGUTI, C. A. Tecnologias para localização interna (*indoor location*). *II RST - Seminário de Redes e Sistemas de Telecomunicações*, INATEL, BR, 2015. ISSN 2358-1913. Citado na página 21.

SADOWSKI, S.; SPACHOS, P. Rssi-based indoor localization with the internet of things. *IEEE Access*, IEEE, v. 6, p. 30149–30161, 2018. Citado 7 vezes nas páginas 21, 22, 26, 28, 30, 31 e 34.

SAHINOGLU, Z.; GEZICI, S.; GÜVENC, I. *Ultra-wideband Positioning Systems: Theoretical Limits, Ranging Algorithms, and Protocols*. Cambridge University Press, 2008. ISBN 9781139472319. Disponível em: <https://books.google.com.br/books?id=ygXzSEzQy4oC>. Acesso em: 20 out. 2018. Citado 6 vezes nas páginas 27, 28, 29, 30, 31 e 32.

SAMIE, F.; BAUER, L.; HENKEL, J. Iot technologies for embedded computing: A survey. In: ACM. *Proceedings of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*. [S.l.], 2016. p. 8. Citado 2 vezes nas páginas 25 e 26.

TEXAS INSTRUMENTS. *Bluetooth low energy Beacons*. [S.l.], 2016. Disponível em: <http://www.ti.com/lit/an/swra475a/swra475a.pdf>. Acesso em: 17 nov. 2018. Citado 2 vezes nas páginas 36 e 37.

XIAO, Y. et al. Full-duplex machine-to-machine communication for wireless-powered internet-of-things. In: . [S.l.]: IEEE, 2016. p. 1–6. ISBN 9781479966646. ISSN 1938-1883. Citado na página 25.

ČABARKAPA, D.; GRUJIć, I.; PAVLOVIć, P. Comparative analysis of the bluetooth low-energy indoor positioning systems. In: *2015 12th International Conference on Telecommunication in Modern Satellite, Cable and Broadcasting Services (TELSIKS)*. [S.l.: s.n.], 2015. p. 76–79. Citado 3 vezes nas páginas 35, 36 e 37.